Stat Comput (2012) 22:1–16
DOI 10.1007/s11222-010-9201-4

Efficient Monte Carlo simulation via the generalized splitting method

Zdravko I. Botev · Dirk P. Kroese

Received: 25 May 2009 / Accepted: 29 August 2010 / Published online: 16 September 2010

Abstract We describe a new Monte Carlo algorithm for the consistent and unbiased estimation of multidimensional integrals and the efficient sampling from multidimensional densities. The algorithm is inspired by the classical splitting method and can be applied to general static simulation models. We provide examples from rare-event probability estimation, counting, and sampling, demonstrating that the proposed method can outperform existing Markov chain sampling methods in terms of convergence speed and accuracy.

Keywords Splitting method · RESTART · MCMC · Rare-event probability estimation · Level-crossing · Convergence diagnostic · Importance sampling · Boolean Satisfiability problem · Fixed effort · Fixed splitting · Sequential Monte Carlo · Combinatorial counting

1 Introduction

One of the first Monte Carlo techniques for rare-event probability estimation is the *splitting method*, proposed by Kahn and Harris (1951). In the splitting technique, sample paths of a Markov process are split into multiple copies at various stages of the simulation, with the objective of generating more occurrences of the rare event. The rare event is represented as the intersection of a nested sequence of events, and the probability of the rare event is thus the product of conditional probabilities, each of which can be estimated much more accurately than the rare-event probability itself. Applications of the splitting method in this Markovian setting arise in particle transmission (Kahn and Harris 1951), queueing systems (Garvels 2000; Garvels and Kroese 1998; Garvels et al. 2002), and reliability (L'Ecuyer et al. 2006). The splitting method has remained popular in the physics and statistics community and has gradually evolved over the past two decades (Glasserman et al. 1996, 1998; Villén-Altamirano and Villén-Altamirano 1991, 1994, 1999), to become an effective simulation technique for dynamic simulation models. More recent improvements include theoretical results about the optimal selection of the splitting levels (Cérou and Guyader 2007; Lagnoux-Renaudie 2006, 2008; Garvels 2000) and the use of quasi Monte Carlo estimators (L'Ecuyer et al. 2007).

The aim of this paper is to introduce a new algorithm, called the *Generalized Splitting* (GS) algorithm, which extends the applicability of the splitting method to both static (that is, time-independent) and non-Markovian models. The GS algorithm has its roots in an adaptive population Monte Carlo method described in Botev (2007, 2008), Cérou et al. (2009) that uses the product rule of probability to estimate rare-event probabilities. In contrast to the specific Markov setting of the classical splitting method, the GS method involves the following general framework (Rubinstein and Kroese 2004): let ℓ be the expected performance of a stochastic system, of the form

$$\ell = \mathbb{E}_f[H(\mathbf{X})] = \int f(\mathbf{x}) H(\mathbf{x}) \mu(d\mathbf{x}), \tag{1}$$

where H is a real-valued function, $\mathbf{X}$ is a random vector, and f the density of $\mathbf{X}$ with respect to some base measure μ. A special case of (1) is obtained when $H(\mathbf{x}) = I\{S(\mathbf{x}) \geq \gamma\}$, where S is a function and γ a parameter large enough such

Z.I. Botev (✉)
Department of Computer Science and Operations Research, University of Montreal, Quebec 6128, Canada
e-mail: botev@maths.uq.edu.au

D.P. Kroese
School of Mathematic and Physics, The University of Queensland, Brisbane 4072, Australia

that

$$\ell = \ell(\gamma) = \mathbb{E}_f[I\{S(\mathbf{X}) \geq \gamma\}] = \mathbb{P}_f(S(\mathbf{X}) \geq \gamma) \tag{2}$$

is very small, so that $\ell(\gamma)$ is a rare-event probability (Rubinstein and Kroese 2004, 2007). Another special case of (1) is obtained when $H(\mathbf{x}) = \mathrm{e}^{-S(\mathbf{x})/\gamma}$, which arises frequently in statistical mechanics in the estimation of the so-called partition function (Robert and Casella 2004).

Using the GS algorithm, we construct unbiased estimators for rare-event probabilities of the form (2)—and, in general, multidimensional integrals of the form (1). In addition, the method provides unbiased estimates for the variances of the estimators. The GS method tackles these static non-Markovian problems by artificially constructing a Markov chain using, for example, Gibbs or Hit-and-Run moves, and then applying the splitting idea to the Markov process induced by these moves.

The GS algorithm has the following advantages over existing MCMC algorithms for estimating (1). First, the GS algorithm provides an unbiased estimator $\widehat{\ell}$ for ℓ in (1) without the need for a burn-in period. In other words, it is not necessary that the Markov chain constructed by the algorithm reach stationarity in order to obtain unbiased and consistent estimates for ℓ. However, a well mixing chain reduces the variance and the simulation effort. Second, unlike most MCMC algorithms, the GS algorithm provides a consistent and unbiased estimate of the mean square error of $\widehat{\ell}$. Third, problems associated with selecting appropriate starting values for the Markov chain in the GS algorithm are strongly mitigated. Finally, while the stationarity of the chain constructed by the GS algorithm is not essential for the estimation of ℓ within the GS framework, testing the hypothesis that the chain has reached stationarity is easy and computationally inexpensive. These properties allow for substantial computational savings over traditional MCMC algorithms.

Many Sequential Monte Carlo (SMC) methods (Johansen et al. 2006; Del Moral et al. 2006) can be viewed as classical splitting algorithms for rare-event simulation, and vice-versa. For example, the fixed effort (FE) splitting first proposed in Garvels (2000), Garvels and Kroese (1998) is a special case of the SMC method in Johansen et al. (2006) with fixed number of particles at each iteration. It follows that all splitting procedures can be significantly enriched by the numerous SMC theoretical results in Del Moral (2004). However, for the static case presented in this paper we have the following essential differences from the fixed effort splitting in Garvels (2000), Garvels and Kroese (1998), Del Moral (2004). All SMC methods use a bootstrap resampling step to replicate the same point in a population numerous times. This typically decreases the diversity of the Monte Carlo population. In contrast, there is no apparent bootstrap resampling step in the proposed GS algorithm. As a result, the performance of the GS estimator is improved due to the increased diversity in the Monte Carlo population, and the unbiasedness property in Proposition 3.1 is not a consequence of similar results in the SMC (Del Moral et al. 2006) or the splitting (Garvels 2000; Garvels and Kroese 1998) literature. In addition, the formulation of the splitting method as a SMC method does not give a concrete implementable algorithm (Del Moral 2004) in the rare-event simulation setting. The SMC framework (Del Moral et al. 2006) does not provide guidance as to how to construct the sequence of importance sampling densities. The GS algorithm described in this paper addresses these issues by selecting the sequence of intermediate distributions adaptively, and makes the connections between SMC and the fixed effort splitting methods more transparent.

The rest of the paper is organized as follows. In Sect. 2 we review the classical splitting method. In Sect. 3 we explain how to obtain unbiased estimates of (2) using the GS methodology. We apply the method to the satisfiability (SAT) counting problem—a combinatorial counting problem. We prove the unbiasedness property of the GS algorithm and explain the differences between the classical splitting method and the GS method. In Sect. 5 we show how SMC (or the equivalent fixed effort splitting) method compares with the GS approach. In Sect. 6 we extend the applicability of the algorithm to the more general problem of estimating (1). In the last Sect. 7 we show how the algorithm can be used for sampling from multidimensional densities for which the standard MCMC methods perform poorly. Finally, we give possible directions for future research.

2 Classical splitting for dynamic simulation

A basic description of the classical splitting method is as follows. Consider a Markov process $X = \{X_u,\, u \geq 0\}$ with state space $\mathcal{X} \subseteq \mathbb{R}^n$, and let $S(\cdot)$ be a real-valued measurable function on $\mathcal{X}$, referred to as the *importance* function. Assume for definiteness that $S(X_0) = 0$. For any *threshold* or *level* $\gamma > 0$, let U_γ denote the first time that the process $S = \{S(X_u),\, u \geq 0\}$ hits the set $[\gamma, \infty)$; and let U_0 denote the first time after 0 that S hits the set $(-\infty, 0]$. We assume that U_γ and U_0 are well-defined finite stopping times with respect to the history of X. Then, one is interested in the probability, ℓ, of the event $E_\gamma = \{U_\gamma < U_0\}$; that is, the probability that S up-crosses level γ before it down-crosses level 0. Note that ℓ depends on the initial distribution of X. The splitting method (Garvels and Kroese 1998; Glasserman et al. 1996) is based on the observation that if $\gamma_2 > \gamma_1$, then $E_{\gamma_2} \subset E_{\gamma_1}$. Therefore, we have by the product rule of probability that $\ell = c_1\, c_2$, with $c_1 = \mathbb{P}(E_{\gamma_1})$ and $c_2 = \mathbb{P}(E_{\gamma_2} \,|\, E_{\gamma_1})$. In many cases, estimation of $c_1 c_2$ by estimating c_1 and c_2 separately is more efficient than the direct Crude Monte Carlo (CMC) estimation of ℓ. Moreover, the same arguments may be used

when the interval $[0, \gamma]$ is subdivided into *multiple* subintervals $[\gamma_0, \gamma_1), [\gamma_1, \gamma_2), \ldots, [\gamma_{T-1}, \gamma]$, where $0 = \gamma_0 < \gamma_1 < \cdots < \gamma_T = \gamma$. Again, let E_{γ_t} denote the event that the process S reaches level γ_t before down-crossing level 0. Since $E_{\gamma_0}, E_{\gamma_1}, \ldots, E_{\gamma_T}$ is a nested sequence of events, denoting $c_t = \mathbb{P}(E_{\gamma_t} \mid E_{\gamma_{t-1}})$, we obtain $\ell = \prod_{t=1}^{T} c_t$.

The estimation of each c_t is performed in the following way. At stage $t = 1$ we start $N_0 \times s_1$ (a fixed number) of independent copies of X and simulate the corresponding process $S(X)$. Each copy of X is run until $S(X)$ either hits the set $(-\infty, 0]$ or up-crosses the level γ_1; that is, each copy is run for a time period equal to $\min\{U_{\gamma_1}, U_0\}$. The number s_1 is an integer referred to as the *splitting factor* at stage $t = 1$. Define I_j^1 to be the indicator that the j-th copy of $S(X)$ hits the set $[\gamma_1, \infty)$ before $(-\infty, 0]$, $j = 1, \ldots, N_0 \times s_1$, and let N_1 be the total number of copies that up-cross γ_1; that is,

$$N_1 = \sum_{j=1}^{N_0 \times s_1} I_j^1.$$

An unbiased estimate for c_1 is $\widehat{c}_1 := \frac{N_1}{N_0 \times s_1}$. For every realization of $S(X)$ which up-crosses γ_1, we store in memory the corresponding state X_τ at the time τ of crossing. Such a state is referred to as the *entrance state* (Garvels 2000). In the next stage, $t = 2$, we start s_2 new independent copies of the chain X from each of the N_1 entrance states, giving a total of $N_1 \times s_2$ new chains. Again, if we let I_j^2 indicate whether the j-th copy of X (starting from an entrance state at level γ_1) hits the set $[\gamma_2, \infty)$ before $(-\infty, 0]$, then $\widehat{c}_2 := \frac{N_2}{N_1 \times s_2}$, where $N_2 = \sum_{j=1}^{N_1 \times s_2} I_j^2$, is an unbiased estimate of c_2, see Garvels (2000). This process is repeated for each subsequent $t = 3, \ldots, T$, such that $N_{t-1} \times s_t$ is the *simulation effort* at stage t, and N_t is the number of entrance states at stage t. The indicators $\{I_j^t\}$ at stage t are usually dependent, and hence the success probabilities $\{\mathbb{P}(I_j^t = 1)\}$ depend on the entrance state from which a copy of the chain X is started. It is well known (Garvels 2000; Glasserman et al. 1996) that despite this dependence, the estimator

$$\widehat{\ell} = \prod_{t=1}^{T} \widehat{c}_t = \frac{N_T}{N_0} \prod_{t=1}^{T} s_t^{-1}$$

is unbiased.

The idea of the splitting method is illustrated in Fig. 1, where three level sets $\{\mathbf{x} : S(\mathbf{x}) = \gamma_t\}$, $t = 0, 1, 2$ are plotted. Here three independent paths of the process $S(X)$ are started from level $\gamma_0 = 0$, two of these paths *die out* by down-crossing level 0, one of the paths up-crosses level γ_1. Three new independent copies of the chain are started from the entrance state at level γ_1 (encircled on the graph), two of these copies down-cross 0, but one copy up-crosses level γ_2. Figure 2 shows a typical realization of a two-dimensional

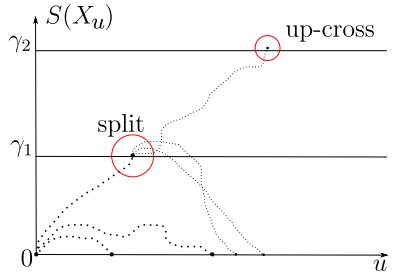

Fig. 1 Typical evolution of the process $S(X)$

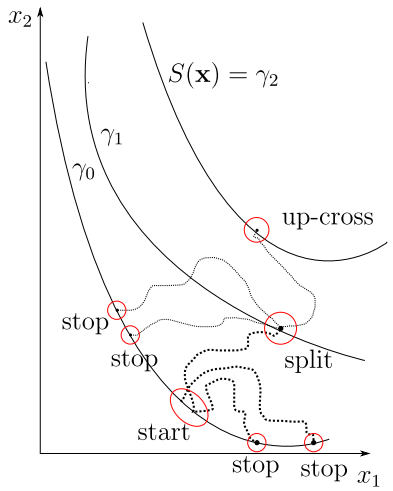

Fig. 2 Typical evolution of the splitting algorithm for a two-dimensional Markov process $\{(X_u^{(1)}, X_u^{(2)}), u \geq 0\}$

Markov process $\{(X_u^{(1)}, X_u^{(2)}), u \geq 0\}$ that corresponds to the scenario described on Fig. 1.

For a given importance function S, the efficiency of the splitting method depends crucially on the number of levels T, the choice of the intermediate levels $\gamma_1, \ldots, \gamma_{T-1}$, and the splitting factors $s_1, \ldots, s_T$. Ideally one would select the levels so that the conditional probabilities $\{c_t\}$ are not too small and easily estimated via CMC. Assuming that the cost of running the Markov process is independent of t, the total simulation effort is a random variable with expected value

$$\begin{aligned}\sum_{t=1}^{T} s_t \mathbb{E}[N_{t-1}] &= N_0 \sum_{t=1}^{T} s_t \ell(\gamma_{t-1}) \prod_{j=1}^{t-1} s_j \\ &= N_0 \sum_{t=1}^{T} s_t \prod_{j=1}^{t-1} c_j s_j \\ &= N_0 \sum_{t=1}^{T} \frac{1}{c_t} \prod_{j=1}^{t} c_j s_j. \end{aligned} \tag{3}$$

An inappropriate choice of the splitting factors may lead to an exponential growth of the simulation effort. For example, if $c_j s_j = a > 1$ for all j, then $N_0 \sum_{t=1}^{T} \frac{1}{c_t} a^t$ grows

exponentially in T. This is referred to as an *explosion* in the splitting literature (Glasserman et al. 1998). Alternatively, if $c_j s_j = a < 1$ for all j, then $\mathbb{E}[N_T] = N_0 a^T$ decays exponentially and with high probability N_T (and hence $\widehat{\ell}$) will be 0, making the algorithm inefficient. It is thus desirable that $c_j s_j = 1$ for all j, that is, the splitting is at *critical value* (Glasserman et al. 1998). In practice, one obtains rough estimates $\{\varrho_j\}$ of $\{c_j\}$ via a pilot run and then initializes from each entrance state $j = 1, \ldots, N_t$, at every stage t, $s_t = \varrho_t^{-1}$ paths. In case $1/\varrho_j$ is not an integer, one can generate a Bernoulli random variable with success probability $\varrho_j^{-1} - \lfloor \varrho_j^{-1} \rfloor$ and then add it to $\lfloor \varrho_j^{-1} \rfloor$ to obtain a random integer-valued splitting factor $\mathcal{S}_j$ with expected value $1/\varrho_j$. This version of the splitting algorithm is called the *Fixed Splitting* (FS) implementation, because at every stage t one generates a fixed expected number of copies ϱ_t^{-1} from each entrance state. An alternative to the FS implementation is the *Fixed Effort* (FE) implementation, where the simulation effort is fixed to N at each stage, instead of the number of copies (Garvels 2000). The estimator then is

$$\widehat{\ell}_{\mathrm{FE}} = \prod_{t=1}^{T} \frac{N_t}{N}.$$

The FE implementation prevents explosions in the number of total Markov chain copies, but has the disadvantage that it is more difficult to analyze the variance of $\widehat{\ell}_{\mathrm{FE}}$ (Garvels 2000; Garvels and Kroese 1998).

3 Splitting for static rare-event probability estimation

We now explain how one can obtain unbiased estimates of the rare-event probability (2) using the splitting idea described in the previous section. First, choose the importance function S and partition the interval $(-\infty, \gamma]$ using intermediate levels $-\infty = \gamma_0 \le \gamma_1 \le \cdots \le \gamma_{T-1} \le \gamma_T = \gamma$. Note that, unlike in the classical splitting, $\gamma_0 = -\infty$. We assume that the sequence of levels is chosen such that the conditional probabilities $\mathbb{P}_f(S(\mathbf{X}) \ge \gamma_t \,|\, S(\mathbf{X}) \ge \gamma_{t-1}) = c_t$, $t = 1, \ldots, T$, are not rare-event probabilities, and that we have rough estimates $\{\varrho_t\}$ of $\{c_t\}$ available. These estimates cannot usually be determined on-line, but we later explain how we can construct the sequence $\{\gamma_t, \varrho_t\}_{t=1}^T$ using the adaptive pilot algorithm described in Botev (2008). Without loss of generality, we assume that generating random variables from f is easy.

Algorithm 3.1 (GS algorithm for estimating $\ell = \mathbb{P}_f(S(\mathbf{X}) \ge \gamma)$) Given a sequence $\{\gamma_t, \varrho_t\}_{t=1}^T$ and a sample size N, execute the following steps.

1. *Initialization*. Set $t = 1$ and $N_0 = \varrho_1 \lfloor \frac{N}{\varrho_1} \rfloor$ (which ensures that N_0/ϱ_1 is an integer). Generate

   $$\mathbf{X}_1, \ldots, \mathbf{X}_{N_0/\varrho_1} \sim_{\mathrm{iid}} f(\mathbf{x})$$

   and denote $\mathcal{X}_0 = \{\mathbf{X}_1, \ldots, \mathbf{X}_{N_0/\varrho_1}\}$. Let $\mathcal{X}_1 = \{\mathbf{X}_1, \ldots, \mathbf{X}_{N_1}\}$ be the largest subset of elements in $\mathcal{X}_0$ for which $S(\mathbf{X}) \ge \gamma_1$ (points $\mathbf{X}_1$ and $\mathbf{X}_2$ on Fig. 3), and let N_1 be the size of $\mathcal{X}_1$. If $N_1 = 0$, go to Step 4.
2. *Markov chain sampling*. For each $\mathbf{X}_i$ in $\mathcal{X}_t = \{\mathbf{X}_1, \ldots, \mathbf{X}_{N_t}\}$, sample independently:

   $$\mathbf{Y}_{ij} \sim \kappa_t(\mathbf{y} \,|\, \mathbf{Y}_{i,j-1}), \qquad \mathbf{Y}_{i,0} = \mathbf{X}_i, \quad j = 1, \ldots, \mathcal{S}_{ti}, \tag{4}$$

   where

   $$\mathcal{S}_{ti} - \left\lfloor \frac{1}{\varrho_{t+1}} \right\rfloor \sim_{\mathrm{iid}} \mathsf{Ber}\left(\frac{1}{\varrho_{t+1}} - \left\lfloor \frac{1}{\varrho_{t+1}} \right\rfloor \right), \quad i = 1, \ldots, N_t.$$

   Here $\kappa_t(\mathbf{y} \,|\, \mathbf{Y}_{i,j-1})$ is a Markov transition density with stationary pdf

   $$f_t(\mathbf{y}) = \frac{f(\mathbf{y}) I\{S(\mathbf{y}) \ge \gamma_t\}}{\ell(\gamma_t)}.$$

   Each $\mathcal{S}_{ti}$ is a splitting factor equal to $\lfloor \frac{1}{\varrho_{t+1}} \rfloor$ plus a Bernoulli random variable with success probability $\frac{1}{\varrho_{t+1}} - \lfloor \frac{1}{\varrho_{t+1}} \rfloor$. Reset

   $$\mathcal{X}_t \equiv \{\mathbf{Y}_{1,1}, \mathbf{Y}_{1,2}, \ldots, \mathbf{Y}_{1,\mathcal{S}_{t,1}}, \ldots, \mathbf{Y}_{N_t,1}, \mathbf{Y}_{N_t,2}, \ldots, \mathbf{Y}_{N_t,\mathcal{S}_{t,N_t}}\},$$

   where $\mathcal{X}_t$ contains $|\mathcal{X}_t| = \sum_{i=1}^{N_t} \mathcal{S}_{ti}$ elements and $\mathbb{E}[|\mathcal{X}_t| \,|\, N_t] = \frac{N_t}{\varrho_{t+1}}$. For example, on Fig. 3 we have $\mathcal{X}_1 = \{\mathbf{X}_{ij},\ i = 1, 2;\ j = 1, \ldots, 10\}$ and $|\mathcal{X}_1| = 20$.
3. *Updating*. Let $\mathcal{X}_{t+1} = \{\mathbf{X}_1, \ldots, \mathbf{X}_{N_{t+1}}\}$ be the largest subset of elements in $\mathcal{X}_t$ for which $S(\mathbf{X}) \ge \gamma_{t+1}$. Here, N_{t+1} is the random number of vectors in $\mathcal{X}_t$ for which $S(\mathbf{X}) \ge \gamma_{t+1}$ (for example, $\mathcal{X}_2 = \{\mathbf{X}_{1,j},\ j = 2, 6, 7\}$ and $N_2 = 3$ on Fig. 3). Reset the level counter $t := t + 1$.
4. *Stopping condition*. If $t = T$ or $N_t = 0$, set $N_{t+1} = N_{t+2} = \cdots = N_T = 0$ and go to Step 5; otherwise, repeat from Step 2.
5. *Final estimator*. Deliver the unbiased estimate of the rare-event probability:

   $$\widehat{\ell} = \frac{N_T}{N_0} \prod_{t=1}^{T} \varrho_t, \tag{5}$$

   and the unbiased estimate of the variance:

   $$\widehat{\mathrm{Var}(\widehat{\ell})} = \frac{\prod_{t=1}^{T} \varrho_t^2}{N_0(N_0 - \varrho_1)} \sum_{i=1}^{N_0/\varrho_1} \left(O_i - \frac{\varrho_1}{N_0} N_T \right)^2, \tag{6}$$

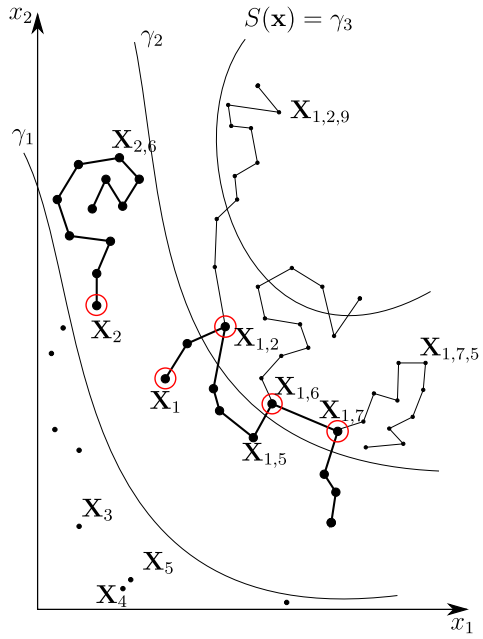

Fig. 3 Typical evolution of the GS algorithm in a two dimensional state space

where O_i denotes the number of points in $\mathcal{X}_T$ that share a common history with the i-th point from the initial population $\mathcal{X}_0$ and are above $\gamma_T = \gamma$ at the final $t = T$ stage. For example, on Fig. 3 we have $O_1 = 11$ and $O_i = 0$ for $i = 2, \ldots, 10$, since only points $\mathbf{X}_{1,2,k}$, $k = 3, \ldots, 10$ and $\mathbf{X}_{1,6,k}$, $k = 7, 8, 10$ are above γ_3 threshold and they are all part of a branch that has $\mathbf{X}_1$ at its root.

In Step 2 of Algorithm 3.1 a move from $\mathbf{X}$ to $\mathbf{Y}$ (using the transition density $\kappa_t(\mathbf{y} \,|\, \mathbf{x})$) can, for example, consist of drawing $\mathbf{Y}$ from the conditional pdf

$$Y_i \sim f_t(y_i \,|\, Y_1, \ldots, Y_{i-1}, X_{i+1}, \ldots, X_n), \quad i = 1, \ldots, n,$$

as in the Gibbs sampling method (Rubinstein and Kroese 2007). The transition density is then

$$\kappa_t(\mathbf{y} \,|\, \mathbf{x}) = \prod_{i=1}^{n} f_t(y_i \,|\, y_1, \ldots, y_{i-1}, x_{i+1}, \ldots, x_n). \tag{7}$$

Alternatively, a move from $\mathbf{X}$ to $\mathbf{Y}$ may consist of a Hit-and-Run move (Chen et al. 2000):

1. Generate a uniformly distributed point on the surface of the n-dimensional hyper-sphere:

$$\mathbf{d} = \left(\frac{Z_1}{\|\mathbf{Z}\|}, \ldots, \frac{Z_n}{\|\mathbf{Z}\|} \right)^{\mathsf{T}}, \quad \mathbf{Z} \sim \mathsf{N}(\mathbf{0}; I).$$

   Here $\|\mathbf{z}\|^2 \stackrel{\text{def}}{=} z_1^2 + \cdots + z_n^2$.
2. Given the current state $\mathbf{X}$, generate Λ from the density:

$$q(\lambda \,|\, \mathbf{X}, \mathbf{d}) \stackrel{\text{def}}{=} \frac{f(\mathbf{X} + \lambda \mathbf{d})}{\int_{-\infty}^{\infty} f(\mathbf{X} + u\mathbf{d})\, \mathrm{d}u}.$$

3. The new state of the chain is:

$$\mathbf{Y} = \begin{cases} \mathbf{X} + \Lambda \mathbf{d}, & \text{if } S(\mathbf{X} + \Lambda \mathbf{d}) \geq \gamma_t, \\ \mathbf{X}, & \text{otherwise.} \end{cases}$$

We illustrate Algorithm 3.1 on a typical problem of the form (2) with three levels ($T = 3$). Figure 3 depicts the GS recipe as applied to a particular two-dimensional rare-event simulation problem. The three level sets $\{\mathbf{x} : S(\mathbf{x}) = \gamma_t\}$, $t = 1, 2, 3$ are plotted as nested curves and the entrance states at each stage are encircled. We assume that the $\{\gamma_t\}$ are given and that $\varrho_t = 1/10$ for all t; that is, the splitting factors are $s_t = \varrho_t^{-1} = 10$ for all t. Initially, at stage $t = 1$, we generate $N_0/\varrho_1 = 10$ independent points from the density $f(\mathbf{x})$. We denote the points $\mathbf{X}_1, \ldots, \mathbf{X}_{10}$. Two of these points, namely $\mathbf{X}_1$ and $\mathbf{X}_2$, are such that both $S(\mathbf{X}_1)$ and $S(\mathbf{X}_2)$ are above the γ_1 threshold. Points $\mathbf{X}_1$ and $\mathbf{X}_2$ are thus the entrance states for the next stage of the algorithm, and $N_1 = \sum_{j=1}^{10} I\{S(\mathbf{X}_j) \geq \gamma_1\} = 2$. In stage $t = 2$ we start independent Markov chains from each of the entrance states $\mathbf{X}_1$ and $\mathbf{X}_2$. The only requirement is that each Markov chain has a stationary distribution equal to the conditional distribution of $\mathbf{X}$ given that $S(\mathbf{X}) \geq \gamma_1$, where $\mathbf{X} \sim f$. The length, or the number of steps, of both chains is equal to $s_2 = 10$. Thus, the simulation effort for $t = 2$ is $N_1 \times s_2 = 20$. In other words, in stage $t = 2$ we generate

$$\mathbf{X}_{i,j} \sim \kappa_1(\mathbf{x} \,|\, \mathbf{X}_{i,j-1}), \quad j = 1, \ldots, 10, \ i = 1, 2,$$

where $\mathbf{X}_{i,0} = \mathbf{X}_i$, and $\kappa_1(\cdot \,|\, \cdot)$ is a Markov transition density with stationary pdf f_1 given by ($t = 1$)

$$f_t(\mathbf{x}) = \frac{f(\mathbf{x}) I\{S(\mathbf{x}) \geq \gamma_t\}}{\ell(\gamma_t)}. \tag{8}$$

Figure 3 depicts the Markov chains as branches sprouting from points $\mathbf{X}_1$ and $\mathbf{X}_2$. Note that these branches are drawn thicker than branches generated at stage $t = 3$. None of the points on the $\mathbf{X}_2$ branch have a value S above γ_2. Points $\mathbf{X}_{1,2}, \mathbf{X}_{1,6}, \mathbf{X}_{1,7}$ from the $\mathbf{X}_1$ branch (encircled) make it above the γ_2 threshold. These three points will be the entrance states for stage $t = 3$ of the algorithm. Thus,

$$N_2 = \sum_{j=1}^{10} \big(I\{S(\mathbf{X}_{1,j}) \geq \gamma_2\} + I\{S(\mathbf{X}_{2,j}) \geq \gamma_2\} \big) = 3.$$

In the final stage we start three independent Markov chains with stationary density f_2 from each of the entrance states $\mathbf{X}_{1,2}, \mathbf{X}_{1,6}, \mathbf{X}_{1,7}$. The length of each chain is $s_3 = 10$. Thus, the simulation effort for stage $t = 3$ is $3 \times 10 = 30$, and we generate

$$\mathbf{X}_{1,j,k} \sim \kappa_2(\mathbf{x} \,|\, \mathbf{X}_{1,j,k-1}), \quad k = 1, \ldots, 10, \ j = 2, 6, 7,$$

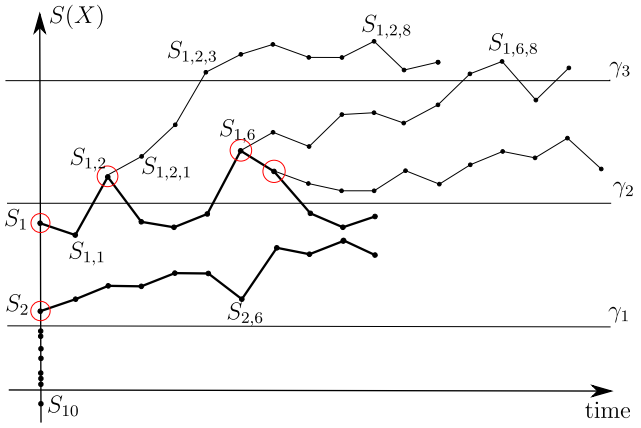

Fig. 4 Typical evolution of $S(\mathbf{X})$ corresponding to the scenario in Fig. 3

where $\mathbf{X}_{1,j,0} = \mathbf{X}_{1,j}$, and $\kappa_2(\cdot\,|\,\cdot)$ is a Markov transition density with stationary density f_2 defined in (8). Figure 3 shows that the points that up-cross level γ_3 are $\mathbf{X}_{1,2,k}$, $k = 3,\ldots,10$ and $\mathbf{X}_{1,6,k}$, $k = 7, 8, 10$. Thus, in the last stage $T = 3$ we have $N_T = 11$. Finally, an estimator of $\ell(\gamma_3)$ is

$$\widehat{\ell}(\gamma_T) = \frac{N_T}{N_0}\prod_{t=1}^{T} s_t^{-1},$$

and this gives the estimate 11×10^{-3}. Proposition 3.1 shows that such an estimator is unbiased. Figure 4 shows the behavior of the importance function process $S(\mathbf{X})$ for every chain starting from the entrance states $\mathbf{X}_1, \mathbf{X}_2$ and $\mathbf{X}_{1,2}, \mathbf{X}_{1,6}, \mathbf{X}_{1,7}$ (encircled). Here $S_{i,j,k} = S(\mathbf{X}_{i,j,k})$ and time is measured in terms of the number of Markov chain moves. Note that the Markov chain paths generated at stage $t = 2$ are drawn thicker. The three chains starting from $\mathbf{X}_{1,2}, \mathbf{X}_{1,6}, \mathbf{X}_{1,7}$ are all dependent, because they share a common history, namely, the branch with root at $\mathbf{X}_1$. These three chains, however, are conditionally independent given the branch with root at $\mathbf{X}_1$.

Example 3.1 (SAT Counting Problem) There are many different mathematical formulations of the Boolean Satisfiability problem (SAT) problem (Gu et al. 1996). Here we use a formulation which is convenient for the problems from the SATLIB website `www.satlib.org`. Let $\mathbf{x} = (x_1,\ldots,x_n)'$, $x_i \in \{0,1\}$ denote a *truth assignment*. Let $A = (A_{ij})$ denote a $m \times n$ *clause matrix*, that is, all elements of A belong to the set $\{-1, 0, 1\}$. Define $\mathbf{b} = (b_1,\ldots,b_m)'$ to be the vector with entries $b_i = 1 - \sum_{j=1}^{n} I\{A_{ij} = -1\}$. In the standard SAT problem one is interested in finding a truth assignment $\mathbf{x}$ for which $A\mathbf{x} \geq \mathbf{b}$. In the SAT *counting* problem, one is interested in finding the *total number* of truth assignments $\mathbf{x}$ for which $A\mathbf{x} \geq \mathbf{b}$. The SAT counting problem is considered more complex than the SAT problem (Rubinstein and Kroese 2007; Welsh 1993), and in fact the SAT counting problem is known to be an #P complete problem.

Table 1 The sequence of levels and splitting factors used in Algorithm 3.1 for instance `uf75-01`

t	γ_t	ϱ_t	t	γ_t	ϱ_t
1	285	0.4750	17	311	0.3072
2	289	0.4996	18	312	0.3104
3	292	0.4429	19	313	0.2722
4	294	0.4912	20	314	0.2253
5	296	0.4369	21	315	0.2235
6	298	0.3829	22	316	0.2071
7	300	0.3277	23	317	0.1892
8	302	0.2676	24	318	0.1722
9	303	0.4982	25	319	0.1363
10	304	0.4505	26	320	0.1413
11	305	0.4359	27	321	0.1237
12	306	0.4239	28	322	0.0953
13	307	0.3990	29	323	0.0742
14	308	0.3669	30	324	0.0415
15	309	0.3658	31	325	0.0115
16	310	0.3166			

Here we aim to find the total number of solutions of the SAT problem, that is, we wish to estimate the size of the set

$$\mathcal{X}^* = \left\{\mathbf{x} : \sum_{i=1}^{m} I\left\{\sum_{j=1}^{n} A_{ij}x_j \geq b_i\right\} \geq m\right\}.$$

To estimate $|\mathcal{X}^*|$ we consider the problem of estimating the probability

$$\begin{aligned} &\ell = \mathbb{P}(S(\mathbf{X}) \geq m),\\ &\{X_j\} \sim_{\text{iid}} \mathsf{Ber}(1/2), \qquad (9)\\ &S(\mathbf{x}) = \sum_{i=1}^{m} I\left\{\sum_{j=1}^{n} A_{ij}x_j \geq b_i\right\}, \end{aligned}$$

via Algorithm 3.1. Thus, each row of A represents a clause and $S(\mathbf{x})$ is the number of clauses that are satisfied. The size of the set is then estimated from the relation $|\mathcal{X}^*| = \ell \times 2^n$. As a numerical example, consider the `uf75-01` problem from the SATLIB website with $m = 325$ and $n = 75$. We applied Algorithm 3.1 with $N = 10^4$ and the splitting factors and levels given in Table 1, giving a total simulation effort of about 2.8×10^6 samples (the expected value is given by (3)), including the pilot run.

The Markov transition density κ_t in Step 2 of Algorithm 3.1 is given by (7) and the stationary pdf is

$$f_t(\mathbf{x}) = \frac{1}{2^n \ell(\gamma_t)} I\left\{\sum_{i=1}^{m} I\left\{\sum_{j=1}^{n} A_{ij}x_j \geq b_i\right\} \geq \gamma_t\right\},$$

$x_j \in \{0,1\}$.

In other words, a move from $\mathbf{x}$ to $\mathbf{y}$ using the density $\kappa_t(\mathbf{y} \mid \mathbf{x})$ is equivalent to the following Gibbs sampling procedure.

1. Given a state $\mathbf{x}$ such that $S(\mathbf{x}) \geq \gamma_t$, generate $Y_1 \sim f_t(y_1 \mid x_2, \ldots, x_n)$;
2. For each $k = 2, \ldots, n-1$, generate $Y_k \sim f_t(y_k \mid Y_1, \ldots, Y_{k-1}, x_{k+1}, \ldots, x_n)$;
3. Finally, generate $Y_n \sim f_t(y_n \mid Y_1, \ldots, Y_{n-1})$.

Note that one can write the conditional density of Y_k as

$$f_t(y_k \mid Y_1, \ldots, Y_{k-1}, x_{k+1}, \ldots, x_n) = \begin{cases} p_k, & y_k = 1, \\ 1 - p_k, & y_k = 0, \end{cases}$$

where $p_k = \frac{I\{s_k^+ \geq \gamma_t\}}{I\{s_k^+ \geq \gamma_t\} + I\{s_k^- \geq \gamma_t\}}$ with $s_k^- = S(Y_1, \ldots, Y_{k-1}, 0, x_{k+1}, \ldots, x_n)$, $s_k^+ = S(Y_1, \ldots, Y_{k-1}, 1, x_{k+1}, \ldots, x_n)$.

With the above setup, we obtain $|\widehat{\mathcal{X}^*}| = 2.31 \times 10^3$ with estimated relative error of 5.8%. Total enumeration of all possible truth assignments for which $A\mathbf{x} \geq \mathbf{b}$ would require the equivalent of a simulation effort of size $2^{75} \approx 3.7 \times 10^{22}$ and is hence impracticable. To achieve the same relative error using CMC would require a sample size of approximately $N = 4.8 \times 10^{21}$. Thus, we see that with a minimal amount of additional work the GS algorithm has reduced the simulation effort by a factor of approximately 10^{15}. Note that a better choice for the importance function S may allow for all conditional probabilities $c_t = \mathbb{P}(S(\mathbf{X}) > \gamma_t \mid S(\mathbf{X}) > \gamma_{t-1})$ to be more or less equal, thus giving a superior estimate. However the optimal choice of the importance function in splitting is still an unresolved problem (Garvels 2000; L'Ecuyer et al. 2007).

We can put a deterministic lower bound on $|\mathcal{X}^*|$. The population $\mathcal{X}_T$ at the final iteration of Algorithm 3.1 is approximately uniformly distributed over the set $\mathcal{X}^*$ and as a result can be used to find some of the distinct solutions of the SAT problem. We ran Algorithm 3.1 10 times with $N = 10^4$ and were able to find 2258 distinct solutions amongst the 10 final populations generated at iteration T. Thus, we conclude that $|\mathcal{X}^*| \geq 2258$.

Finally, we note that significant variance reduction can be achieved when the GS algorithm is combined with the Cross Entropy method (Botev 2009). Table 2 shows the estimates with their respective relative error obtained using the combined GS importance sampling estimator described in Botev (2008, 2009). The SAT instances are from Hoos and Stützle (2000).

Concerning the properties of the estimator (5) and its variance, we have the following result.

Proposition 3.1 (Unbiasedness of the GS estimator) *The estimator in* (5) *is an unbiased estimator of* ℓ, *and* (6) *is an unbiased estimator of* $\mathrm{Var}(\widehat{\ell})$.

Table 2 Twelve SAT counting problems with the current best estimate obtained via the combined GS importance sampling estimator described in Botev (2009)

Case number	SAT Instance	$\widehat{\lvert\mathcal{X}^*\rvert}$	Relative error
1	uf75-01	2258.28	0.03%
2	uf75-02	4590.02	0.07%
3	uf250-01	3.38×10^{11}	4.4%
4	RTI_k3_n100_m429_0	20943.79	0.01%
5	RTI_k3_n100_m429_1	24541.70	0.02%
6	RTI_k3_n100_m429_2	3.9989	0.01%
7	RTI_k3_n100_m429_3	376.016	0.01%
8	RTI_k3_n100_m429_4	4286.28	0.3%
9	RTI_k3_n100_m429_5	7621.11	0.7%
10	RTI_k3_n100_m429_6	2210.20	0.01%
11	RTI_k3_n100_m429_7	1869.64	0.3%
12	RTI_k3_n100_m429_8	1832.29	0.01%

Proof Using the notation of Fig. 3, we can write

$$N_T = \sum_{\mathbf{p}} \prod_{t=1}^{T} I\{S(\mathbf{X}_{p_1,\ldots,p_t}) \geq \gamma_t\},$$

where $\mathbf{p} = (p_1, \ldots, p_T)$, and in the sum above p_1 ranges over $1, \ldots, N_0/\varrho_1$ and p_t, $t \geq 2$, ranges over $1, \ldots, \mathcal{S}_{t-1,p_{t-1}}$. In addition, $\mathbf{X}_{p_1} \sim f(\cdot)$, independently for all p_1, and for $t \geq 2$ we have $\mathbf{X}_{p_1,\ldots,p_t} \sim \kappa_{t-1}(\cdot \mid \mathbf{X}_{p_1,\ldots,p_t-1})$ with $\mathbf{X}_{p_1,\ldots,p_{t-1},0} = \mathbf{X}_{p_1,\ldots,p_{t-1}}$.

Since the splitting factors $\{\mathcal{S}_{t,p_t}\}$ are independent of $\{\mathbf{X}_{p_1}, \mathbf{X}_{p_1 p_2}, \ldots, \mathbf{X}_{\mathbf{p}}\}$, we can write

$$\mathbb{E}[N_T \mid \{\mathcal{S}_{t,p_t}\}] = \sum_{\mathbf{p}} \mathbb{E} \prod_{t=1}^{T} I\{S(\mathbf{X}_{p_1,\ldots,p_t}) \geq \gamma_t\}.$$

The expectation under the multiple summation is

$$\int \cdots \int f(\mathbf{x}_{p_1}) I_{\{S(\mathbf{x}_{p_1}) \geq \gamma_1\}} \times \prod_{t=2}^{T} \prod_{l=1}^{p_t} \kappa_{t-1}(\mathbf{x}_{p_1,\ldots,p_{t-1},l} \mid \mathbf{x}_{p_1,\ldots,p_{t-1},l-1}) \times I\{S(\mathbf{x}_{p_1,\ldots,p_t}) \geq \gamma_t\}\, d\mathbf{x}_{p_1} d\mathbf{x}_{p_1,\ldots,p_t},$$

which by integrating in the order defined by $d\mathbf{x}_{p_1} d\mathbf{x}_{p_1,\ldots,p_t}$ and each time applying the invariance property

$$\int f(\mathbf{x}) I\{S(\mathbf{x}) \geq \gamma_t\} \kappa_t(\mathbf{y} \mid \mathbf{x})\, d\mathbf{x} = f(\mathbf{y}) I\{S(\mathbf{y}) \geq \gamma_t\}, \quad \text{for all } t, \tag{10}$$

yields $\ell = \ell(\gamma_T) = \int f(\mathbf{x}) I\{S(\mathbf{x}) \geq \gamma_T\} d\mathbf{x}$. Therefore

$$\mathbb{E}[N_T] = \mathbb{E}[\mathbb{E}[N_T \mid \{\mathcal{S}_{t,p_t}\}]] = \ell \mathbb{E}\left[\sum_{\mathbf{p}} 1\right] = \ell \frac{N_0}{\prod_{t=1}^{T} \varrho_t},$$

and the estimator (5) is unbiased. To derive the variance of (5), observe that by definition

$$O_{p_1} = I\{S(\mathbf{X}_{p_1}) \geq \gamma_1\} \sum_{p_2,\ldots,p_T} \prod_{t=2}^{T} I\{S(\mathbf{X}_{p_1,\ldots,p_t}) \geq \gamma_t\}.$$

Since the $\{\mathbf{X}_{p_1}\}$ are independent and $N_T = \sum_{p_1} O_{p_1}$, we have $\mathrm{Var}(N_T) = \frac{N_0}{\varrho_1} \mathrm{Var}(O_{p_1})$, from which (6) follows. □

Whatever the mixing speed of the Markov chains, the estimator $\widehat{\ell}$ is unbiased. However, that does not mean that the mixing of the chain is irrelevant. In the extreme case of no mixing at all, that is, when the chain does not move in the sample space, the estimator $\widehat{\ell}$ reduces to the unbiased Crude Monte Carlo estimator of the rare-event probability ℓ.

4 An adaptive generalized splitting algorithm

We now describe the algorithm which we use as a pilot run to estimate the splitting factors $\{\varrho_t\}$ and the levels $\{\gamma_t\}$. It is the earliest version of the GS algorithm (Botev 2008), and we will refer to it as the ADAM algorithm, which stands for ADAptive Multilevel splitting algorithm. For example, Table 1 is created using Algorithm 4.1 with $N = 1000$, $\varrho = 0.5$, and the Markov transition density in Example 3.1.

Algorithm 4.1 (ADAM Algorithm) Given the sample size N and the parameters $\varrho \in (0, 1)$ and γ, execute the following steps.

1. *Initialization*. Set the counter $t = 1$.
   (a) Generate $\mathbf{X}_1, \ldots, \mathbf{X}_N \sim f(\mathbf{x})$ and denote $\mathcal{X}_0 = \{\mathbf{X}_1, \ldots, \mathbf{X}_N\}$.
   (b) Let

$$\widetilde{\gamma}_t = \underset{\gamma \in \{S_1,\ldots,S_N\}}{\operatorname{argmin}} \left\{ \frac{1}{N} \sum_{i=1}^{N} I\{S(\mathbf{X}_i) \geq \gamma\} \leq \varrho \right\}, \quad \mathbf{X}_i \in \mathcal{X}_{t-1}. \tag{11}$$

   That is, $\widetilde{\gamma}_t$ is the smallest value from amongst $S(\mathbf{X}_1), \ldots, S(\mathbf{X}_N)$ such that $\frac{1}{N}\sum_{i=1}^{N} I\{S(\mathbf{X}_i) \geq \widetilde{\gamma}_t\} \leq \varrho$. Set $\gamma_t = \min\{\gamma, \widetilde{\gamma}_t\}$. Let $\mathcal{X}_t = \{\mathbf{X}_1, \ldots, \mathbf{X}_{N_t}\}$ be the largest subset of elements in $\mathcal{X}_{t-1}$ for which $S(\mathbf{X}) \geq \gamma_t$. Let $N_t = |\mathcal{X}_t|$. Then, $\varrho_t = \frac{N_t}{N}$ is an approximation of the probability $c_t = \mathbb{P}_f(S(\mathbf{X}) \geq \gamma_t \mid S(\mathbf{X}) \geq \gamma_{t-1})$, $\gamma_0 = -\infty$.

2. *Markov chain sampling*. Identical to Step 2 of Algorithm 3.1, except that in (4) the splitting factors are generated in a different way, namely,

$$\mathcal{S}_{ti} = \left\lfloor \frac{N}{N_t} \right\rfloor + B_i, \quad i = 1, \ldots, N_t.$$

   Here each $B_1, \ldots, B_{N_t}$ are $\mathsf{Ber}(1/2)$ random variables conditional on $\sum_{i=1}^{N_t} B_i = N \mod N_t$. More precisely, $(B_1, \ldots, B_{N_t})$ is a binary vector with joint pdf

$$\mathbb{P}(B_1 = b_1, \ldots, B_{N_t} = b_{N_t}) = \frac{(N_t - r)!\, r!}{N_t!} I\{b_1 + \cdots + b_{N_t} = r\}, \quad b_i \in \{0, 1\},$$

   where $r = N \mod N_t$. As a consequence of the different generation of the splitting factors, after resetting

$$\mathcal{X}_t \equiv \{\mathbf{Y}_{11}, \mathbf{Y}_{12}, \ldots, \mathbf{Y}_{1,\mathcal{S}_{t1}}, \ldots, \mathbf{Y}_{N_t 1}, \mathbf{Y}_{N_t 2}, \ldots, \mathbf{Y}_{N_t,\mathcal{S}_{tN_t}}\},$$

   the set $\mathcal{X}_t$ contains exactly N elements.
3. *Updating and Estimation*. Reset the counter $t := t + 1$ and proceed exactly as in part (b) of Step 1.
4. *Stopping condition*. If $\gamma_t = \gamma$, set $T = t$ and go to Step 5; otherwise, repeat from Step 2.
5. *Final estimates*. Deliver the estimated levels $\gamma_1, \ldots, \gamma_T$, the splitting factors $\varrho_1, \ldots, \varrho_T$, and the (biased) estimate of the rare-event probability:

$$\widehat{\ell}_{\mathsf{ADAM}} = \prod_{t=1}^{T} \varrho_t = \frac{\prod_{t=1}^{T} N_t}{N^T}. \tag{12}$$

The main differences between the ADAM algorithm and the GS algorithm are the following. First, the difference in Step 2 of the ADAM algorithm is that the splitting factors are generated in a way that fixes the simulation effort at each stage to be N, similar to the fixed effort splitting in Garvels and Kroese (1998). Second, as seen from (11), the levels $\{\gamma_t\}$ are determined online using the random populations $\{\mathcal{X}_t\}$. As a consequence of these differences, the estimator $\widehat{\ell}_{\mathsf{ADAM}}$ is not unbiased and the algorithm does not provide a simple estimate for the variance of $\widehat{\ell}_{\mathsf{ADAM}}$.

The ADAM algorithm can be used as a stand-alone algorithm in the sense that it can provide an estimate of ℓ without the need for any preliminary simulation. For many medium scale problems we could not detect any substantive difference in the numerical performance of Algorithm 4.1 (ADAM) versus Algorithm 3.1 (GS). For example, for the cost of 3.1 million samples ($N = 10^5$, $\varrho = 0.5$) Algorithm 4.1 gave an estimate of $|\widehat{\mathcal{X}^*}| = 2.26 \times 10^3$ with estimated relative error of 3%. However, for large scale problems we detected some bias in the estimates provided by Algorithm 4.1, and the estimated relative error (from repeated

independent runs of ADAM) seemed to underestimate the true relative error. We thus recommend using the GS algorithm (with ADAM used for the pilot run) instead of using ADAM by itself, because the GS algorithm gives provably unbiased estimates for ℓ and the variance of $\widehat{\ell}$. We refer to Cérou et al. (2009) for some results and discussion about the asymptotic bias.

5 Fixed effort generalized splitting

Recall that the GS algorithm is a generalization of the classical Fixed Splitting (FS) algorithm described in the introduction and presented in Garvels and Kroese (1998). It is possible to develop a generalized version of the FE splitting method, so that the possibility of population explosions does not exist. Since the FE splitting in Garvels and Kroese (1998) can be interpreted as a particular SMC algorithm (Johansen et al. 2006) (with systematic resampling), the following algorithm can also be interpreted as an SMC algorithm.

Algorithm 5.1 (Fixed Effort Generalized Splitting) Given a sequence $\{\gamma_t\}_{t=1}^T$ and sample size N, execute the following steps.

1. *Initialization*. Set the counter $t = 1$.
   (a) Generate $\mathbf{X}_1, \ldots, \mathbf{X}_N \sim_{\text{iid}} f(\mathbf{x})$ and let $\mathcal{X}_0 = \{\mathbf{X}_1, \ldots, \mathbf{X}_N\}$.
   (b) Let $\mathcal{X}_t = \{\mathbf{X}_1, \ldots, \mathbf{X}_{N_t}\}$ be the largest subset of elements in $\mathcal{X}_{t-1}$ for which $S(\mathbf{X}) \geq \gamma_t$, where $N_t = |\mathcal{X}_t|$. If $N_t = 0$, go to Step 4.
2. *Markov chain sampling*. For each $\mathbf{X}_i$ in $\mathcal{X}_t = \{\mathbf{X}_1, \ldots, \mathbf{X}_{N_t}\}$, sample independently:

$$\mathbf{Y}_{i,j} \sim \kappa_t(\mathbf{y} \,|\, \mathbf{X}_i), \quad j = 1, \ldots, \mathcal{S}_{ti}, \tag{13}$$

   where $\mathcal{S}_{ti} = \lfloor \frac{N}{N_t} \rfloor + B_i$, $i = 1, \ldots, N_t$, and $\kappa_t(\mathbf{y} \,|\, \mathbf{X}_i)$ is a Markov transition density with stationary pdf $f_t(\mathbf{y}) = \frac{f(\mathbf{y}) I\{S(\mathbf{y}) \geq \gamma_t\}}{\ell(\gamma_t)}$, and each B_i is a $\mathsf{Ber}(1/2)$ random variable conditional on $\sum_{i=1}^{N_t} B_i = N \mod N_t$. Reset

$$\mathcal{X}_t = \{\mathbf{Y}_{1,1}, \mathbf{Y}_{1,2}, \ldots, \mathbf{Y}_{1,\mathcal{S}_{t1}}, \ldots, \mathbf{Y}_{N_t,1}, \mathbf{Y}_{N_t,2}, \ldots, \mathbf{Y}_{N_t,\mathcal{S}_{tN_t}}\},$$

   where $\mathcal{X}_t$ contains $|\mathcal{X}_t| = \sum_{i=1}^{N_t} \mathcal{S}_{ti} = N$ elements.
3. *Updating*. Reset the counter $t = t + 1$ and proceed exactly as in part (b) of Step 1.
4. *Stopping condition*. If $t = T$ or $N_t = 0$, set $N_{t+1} = N_{t+2} = \cdots = N_T = 0$ and go to Step 5; otherwise, repeat from Step 2.
5. *Final estimator*. Deliver the unbiased estimate $\widehat{\ell}_{\mathsf{FE}} = \frac{1}{N^T} \prod_{t=1}^T N_t$.

We call Algorithm 5.1 the Fixed Effort Generalized Splitting (FE-GS) algorithm to distinguish it from the GS Algorithm 3.1. The main difference between the FE-GS and GS algorithms is in Step 4, and in particular (13) and (4). In FE-GS the $\{\mathcal{S}_{ti}\}$ in (13) depend on the random variable N_t. Thus, the possibility of explosion is avoided by making the splitting dependent on the history of the process. The simulation effort at each stage t is fixed to be $\sum_{i=1}^{N_t} \mathcal{S}_{ti} = N$. Furthermore, the $\{\mathbf{Y}_{i,j}\}$ are sampled by restarting the Markov transition density κ_t from the *same* point $\mathbf{X}_i$. In contrast, in the GS algorithm the $\{\mathcal{S}_{ti}\}$ in (4) are completely independent of $\{N_t\}$ and the past performance of the algorithm. In addition, each $\mathbf{Y}_{i,j}$ is generated from $\kappa_t(\mathbf{y} \,|\, \mathbf{Y}_{i,j-1})$ instead of $\kappa_t(\mathbf{y} \,|\, \mathbf{X}_i)$, thus decreasing the dependence between, say, $\mathbf{Y}_{i,1}$ and $\mathbf{Y}_{i,N_t}$, and giving a more reliable estimate in Step 5. Another advantage of the GS algorithm is the availability of an estimate of the relative error from a single simulation run. To estimate $\mathrm{Var}(\widehat{\ell}_{\mathsf{FE}})$ we have to run the FE-GS algorithm a number of times independently.

Similar to classical FE splitting, the FE-GS estimator is unbiased. For a simple proof see Botev (2009) and for a proof in the more general SMC setting see Johansen et al. (2006).

Example 5.1 (Numerical Comparison Between GS and FE-GS) Consider estimating the probability $\ell(n) = \mathbb{P}(\sum_{i=1}^n X_i \geq n)$, $X_1, \ldots, X_n \sim_{\text{iid}} \mathsf{Ber}(1/2)$. We compare the performance of the FE-GS and GS algorithms on this simple problem using the same simulation effort for both algorithms, and the levels and splitting factors $\{\gamma_t, \varrho_t\}$ determined from a single common run of ADAM with $\varrho = 0.1$ and $\tilde{N} = 10^4$. Table 3 shows the results of the simulation experiment. The first column shows the simulation effort used for 10 independent runs of the FE-GS method (so that each run incurs 1/10 of the total simulation effort). The GS

Table 3 Comparison between GS (fixed splitting) and FE-GS (fixed effort) implementations. The relative error of $\widehat{\ell}(n)$ is given in the brackets

Total simulation effort	n	$\widehat{\ell}(n)$	$\sqrt{\mathrm{Var}(\widehat{\ell}(n))/\mathrm{Var}(\widehat{\ell}_{\mathsf{FE}}(n))}$
10^6	20	9.6×10^{-7} (3%)	0.26
10^6	30	9.1×10^{-10} (2%)	0.20
10^6	40	9.1×10^{-13} (3%)	0.02
10^6	50	9.1×10^{-13} (3%)	0.02
10^6	50	9.4×10^{-16} (4%)	0.03
10^6	60	9.3×10^{-19} (4%)	FE-GS fails
10^6	70	8.9×10^{-22} (5%)	FE-GS fails
10^7	80	8.5×10^{-25} (2%)	FE-GS fails
10^7	90	8.2×10^{-28} (2%)	FE-GS fails
10^7	100	7.7×10^{-31} (2%)	FE-GS fails

Table 4 Comparative performance between the FE-GS and the GS algorithms on the SAT counting problem. The instances are the same as in Table 2

Case	1	2	3	4	5	6	7	8	9	10	11	12
$\frac{\sqrt{\mathrm{Var}(\widehat{\ell})}}{\sqrt{\mathrm{Var}(\widehat{\ell}_{\mathsf{FE}})}}$	0.53	0.13	0.10	0.15	0.29	0.05	0.28	0.04	0.26	0.23	0.62	0.42

method uses a slightly smaller (random) simulation effort. The third column shows the GS estimate $\widehat{\ell}(n)$ with the estimated relative error. The last column shows the ratio of the relative error between $\widehat{\ell}(n)$ (GS estimator) and $\widehat{\ell}_{\mathsf{FE}}(n)$ (FE-GS estimator). The variance of $\widehat{\ell}_{\mathsf{FE}}(n)$ is estimated from the ten independent runs. A value of 0.1 means that for the same simulation effort the GS estimator $\widehat{\ell}(n)$ gives a relative error which is 10% of the relative error of the FE-GS estimator $\widehat{\ell}_{\mathsf{FE}}(n)$. The table suggests that the variance of the FE-GS estimator can be as much as 5000 times the variance of the GS estimator. More importantly, the FE-GS yields an infinite relative error for $n \geq 60$, because the population of points becomes extinct before reaching the final level ($\gamma_T = n$). Increasing the simulation effort to 10^7 did not help the FE-GS algorithm.

As another numerical example we consider the SAT counting instances from Table 2. In all cases the simulation effort for both the FE-GS and GS is kept the same and such that $N = 10^5$ for the GS algorithm and with $\{\gamma_t, \varrho_t\}$ determined by the ADAM algorithm with $\varrho = 0.1$ and $\tilde{N} = 10^4$. The second row of Table 4 shows the ratio $(\mathrm{Var}(\widehat{\ell})/\mathrm{Var}(\widehat{\ell}_{\mathsf{FE}}))^{1/2}$. It can be seen that the GS algorithm performs significantly better than the FE-GS algorithm.

As mentioned in the introduction, the GS algorithm performs better than the FE-GS algorithm, because there is no apparent bootstrap resampling step in the GS algorithm. In summary, we recommend using the ADAM or GS algorithms, instead of the FE-GS algorithm.

6 Estimation of integrals of the form $\mathbb{E}_p[H(\mathbf{Z})]$

In the last section we show how one can estimate rare-event probabilities of the form (2) using either the ADAM or GS. In this section we extend the applicability of these algorithms to the more general problem of estimating integrals of the form (1). To this end we rewrite (1) using the following notation:

$$\mathcal{Z} = \mathbb{E}_p[H(\mathbf{Z})] = \int p(\mathbf{z})\, H(\mathbf{z})\, \mathrm{d}\mathbf{z}, \tag{14}$$

so that now the aim is to estimate $\mathcal{Z}$. We show that the GS algorithm provides an unbiased estimate of $\mathcal{Z}$. First, note that $\mathbb{E}_p[H(\mathbf{Z})] = 2\mathbb{E}_p[H(\mathbf{Z}) I_{\{H(\mathbf{Z})>0\}}] - \mathbb{E}_p[|H(\mathbf{Z})|]$, so that without loss of generality we can consider estimating (14) for $H(\mathbf{z}) \geq 0$. Second, let $\widetilde{p}(\mathbf{z})$ be a proposal density from which it is easy to sample and which dominates $p(\mathbf{z})H(\mathbf{z})$ in the sense that

$$p(\mathbf{z})H(\mathbf{z}) \leq \mathrm{e}^{a\gamma+b}\, \widetilde{p}(\mathbf{z}), \quad \text{for all } \mathbf{z} \in \mathbb{R}^n, \tag{15}$$

for some $\gamma \geq (\ln(\mathcal{Z}) - b)/a$, where $a > 0$ and $b \in \mathbb{R}$ are fixed arbitrary constants. Note that the constant $\mathrm{e}^{a\gamma+b}$ is an upper bound on $\mathcal{Z}$. Typically we have $\mathrm{e}^{a\gamma+b} \gg \mathcal{Z}$ or $\gamma \gg (\ln(\mathcal{Z}) - b)/a$ and in many cases it is natural to choose $\widetilde{p} \equiv p$.

Algorithm 6.1 (Estimation of $\mathcal{Z}$)

1. *Inputs*. Suppose we are given a proposal $\widetilde{p}(\mathbf{z})$, parameters γ, a, b such that (15) holds, and algorithm A. Here A denotes either the GS, or the ADAM algorithm.
2. *Estimation of* ℓ. Use algorithm A to estimate

$$\ell(\gamma) = \mathbb{E}_f[I\{S(\mathbf{X}) \geq \gamma\}] = \int f(\mathbf{x}) I\{S(\mathbf{x}) \geq \gamma\} d\mathbf{x}, \tag{16}$$

   where the vector $\mathbf{x} = (\mathbf{z}, u)' \in \mathbb{R}^n \times [0,1]$ augments $\mathbf{z}$ with the variable $u \in [0,1]$, the value $S(\mathbf{x})$ is given by $S(\mathbf{x}) = \frac{1}{a}\ln(\frac{p(\mathbf{z})H(\mathbf{z})}{u\,\widetilde{p}(\mathbf{z})}) - \frac{b}{a}$, and the density $f(\mathbf{x})$ by $f(\mathbf{x}) = \widetilde{p}(\mathbf{z}) \times I\{0 \leq u \leq 1\}, \quad \mathbf{x} \in \mathbb{R}^{n+1}$.
3. *Estimation of* $\mathcal{Z}$. An estimate of $\mathcal{Z}$ in (14) is: $\widehat{\mathcal{Z}} = \mathrm{e}^{a\gamma+b}\, \widehat{\ell}(\gamma)$.

The following proposition shows that the estimate $\widehat{\mathcal{Z}}$ is unbiased if A is the GS algorithm.

Proposition 6.1 (Relation between ℓ and $\mathcal{Z}$) *The pdf*

$$\frac{p(\mathbf{z})H(\mathbf{z})}{\mathcal{Z}} \tag{17}$$

is the marginal density of $f_T(\mathbf{x}) = \frac{1}{\ell(\gamma_T)} f(\mathbf{x}) I\{S(\mathbf{x}) \geq \gamma_T\}$ *(recall that* $\gamma_T = \gamma$*), and* $\mathcal{Z} = \mathrm{e}^{a\gamma+b}\ell(\gamma)$.

Proof Note that u is an *auxiliary variable* similar to the one used in the Accept-Reject method for random variable generation (Robert and Casella 2004). From (15) it follows that (with $\mathbf{x} = (\mathbf{z}, u)'$ so that $x_{n+1} = u$):

$$
\begin{aligned}
&\int_{\mathbb{R}} f_T(\mathbf{x})\, dx_{n+1} \\
&= \int_{\mathbb{R}} \frac{\widetilde{p}(\mathbf{z}) I\{0 \le u \le 1\}}{\ell(\gamma)} I \\
&\quad \times \left\{ \frac{1}{a} \ln\left(\frac{p(\mathbf{z}) H(\mathbf{z})}{u \widetilde{p}(\mathbf{z})} \right) - \frac{b}{a} \ge \gamma \right\} du \\
&= \frac{\widetilde{p}(\mathbf{z})}{\ell(\gamma)} \int_0^1 I\left\{ u \le \frac{p(\mathbf{z}) H(\mathbf{z})}{\mathrm{e}^{a\gamma+b} \widetilde{p}(\mathbf{z})} \right\} du \\
&= \frac{p(\mathbf{z}) H(\mathbf{z})}{\ell(\gamma) \mathrm{e}^{a\gamma+b}},
\end{aligned}
$$

because $\frac{p(\mathbf{z})H(\mathbf{z})}{\mathrm{e}^{a\gamma+b}\widetilde{p}(\mathbf{z})} \le 1$ for all $\mathbf{z}$ by (15). Since $\mathcal{Z}$ is the normalizing constant of $p(\mathbf{z})H(\mathbf{z})$, we conclude that $\mathcal{Z} = \mathrm{e}^{a\gamma+b}\ell(\gamma)$. □

Example 6.1 (Two Humps Function) Consider the problem of estimating without bias the normalizing constant $\mathcal{Z} = \mathcal{Z}(\lambda)$ of the pdf proportional to

$$
h(\mathbf{z}) = \exp\left(-\frac{z_1^2 + z_2^2 + (z_1 z_2)^2 - 2\lambda z_1 z_2}{2} \right), \quad \mathbf{z} \in \mathbb{R}^2,
$$

for some parameter $\lambda \in \mathbb{R}$, say $\lambda = 12$. The density $h(\mathbf{z})/\mathcal{Z}(12)$ is plotted on the left panel of Fig. 6. This is a problem of the form (14) with $p(\mathbf{z}) = \frac{1}{2\pi}\exp(-\frac{z_1^2+z_2^2}{2})$ and $H(\mathbf{z}) = 2\pi \exp(-\frac{(z_1 z_2)^2 - 2\lambda z_1 z_2}{2})$. Let $\widetilde{p}(\mathbf{z}) = p(\mathbf{z})$, $a = \frac{1}{2}$, $b = \frac{\lambda^2}{2} + \ln(2\pi)$, and A be the GS algorithm in Step 1 of Algorithm 6.1. Then, (16) can be written as $\ell(\gamma) = \mathbb{P}_f(-(Z_1 Z_2 - \lambda)^2 - 2\ln(U) \ge \gamma)$, where the vector $\mathbf{x} = (\mathbf{z}, u)'$ is augmented such that

$$
f(\mathbf{x}) = \frac{1}{2\pi} \exp\left(-\frac{z_1^2 + z_2^2}{2} \right) \times I\{0 < u < 1\},
$$

and the level $\gamma = 0$ is such that (15) holds. To apply the GS algorithm for the estimation of (16), we need to specify the transition pdf κ_t with stationary density

$$
f_t(\mathbf{x}) = \frac{f(\mathbf{x}) I\{-(z_1 z_2 - \lambda)^2 - 2\ln(u) \ge \gamma_t\}}{\ell(\gamma_t)}.
$$

A move from $\mathbf{X}$ to $\mathbf{X}^*$ using the transition density $\kappa_t(\mathbf{X}^* \mid \mathbf{X})$ consists of the following (systematic) Gibbs sampling procedure.

Algorithm 6.2 (Transition density $\kappa_t(\mathbf{X}^* \mid \mathbf{X})$)

1. Given a state $\mathbf{X} = (Z_1, Z_2, U)'$ for which $S(\mathbf{X}) \ge \gamma_t$, generate $Z_1^* \sim f_t(z_1^* \mid Z_2, U)$; that is, draw Z_1^* from a truncated standard normal density on the interval $[I_1, I_2]$, where $I_1 = \min\{\frac{\lambda-\mu}{Z_2}, \frac{\lambda+\mu}{Z_2}\}$, $I_2 = \max\{\frac{\lambda-\mu}{Z_2}, \frac{\lambda+\mu}{Z_2}\}$, and $\mu = \sqrt{-\gamma_t - 2\ln(U)}$.

Table 5 Levels and splitting factors used to compute the normalizing constant of $h(\mathbf{z})$

t	1	2	3	4	5	6
γ_t	−117.91	−77.03	−44.78	−20.18	−4.40	0
ϱ_t	0.1	0.1	0.1	0.1	0.1	0.2853

2. Generate $Z_2^* \sim f_t(z_2^* \mid Z_1^*, U)$; that is, draw Z_2^* from a truncated standard normal density on the interval $[I_1, I_2]$, where $I_1 = \min\{\frac{\lambda-\mu}{Z_1^*}, \frac{\lambda+\mu}{Z_1^*}\}$, $I_2 = \max\{\frac{\lambda-\mu}{Z_1^*}, \frac{\lambda+\mu}{Z_1^*}\}$, and $\mu = \sqrt{-\gamma_t - 2\ln(U)}$.
3. Generate $U^* \sim f_t(u \mid Z_1^*, Z_2^*)$, that is, draw a uniform random variable U^* on the interval $[0, \mu]$, where

$$
\mu = \min\left\{ 1, \exp\left(-\frac{\gamma_t + (Z_1^* Z_2^* - \lambda)^2}{2} \right) \right\}.
$$

Output $\mathbf{X}^* = [Z_1^*, Z_2^*, U^*]$.

To estimate ℓ we apply the GS algorithm with $N = 2000$, $\lambda = 12$, and the levels and splitting factors in Table 5.

We obtain $\widehat{\ell}(\gamma) = 2.92 \times 10^{-6}$ with estimated relative error of 5%. Hence, in Step 3 of Algorithm 6.1 we obtain $\widehat{\mathcal{Z}} = \widehat{\ell}(\gamma) \times \mathrm{e}^{a\gamma+b} = \widehat{\ell}(\gamma) \times 2\pi \mathrm{e}^{\lambda^2/2} = 3.41 \times 10^{26}$ with a relative error of 5%. Note that Table 5 is computed using the ADAM algorithm with $\varrho = 0.1$, $N = 2000$ and using the same transition density κ_t. The combined simulation effort of the GS algorithm and the ADAM algorithm is about 1.2×10^5 samples. In contrast, CMC estimation of $\mathcal{Z}$ via the estimator $\frac{1}{M}\sum_{i=1}^M H(\mathbf{Z}_i)$, $\{\mathbf{Z}_i\}_{\text{iid}} \sim p(\mathbf{z})$, $M = 1.2 \times 10^5$ gives an estimate of 1.6×10^{26} with estimated relative error of 60%.

For this two-dimensional example we are able to verify the simulation results using deterministic quadrature. An approximate value $\mathcal{Z}(12) \approx 3.5390 \times 10^{26}$ is obtained using the deterministic recursive Simpson's rule (Gander and Gautschi 2000). The constant $\mathcal{Z}$ in the next example, however, cannot be easily computed using an alternative method due to the high-dimensionality of the problem.

Example 6.2 (Rosenbrock function) Consider computing the normalizing constant of the pdf proportional to

$$
h(\mathbf{z}) = \exp(-\lambda R(\mathbf{z})), \quad z_i \in [-2, 2],\ i = 1, \ldots, n,
$$

where $R(\mathbf{z}) = \sum_{i=1}^{n-1}(100(z_{i+1} - z_i^2)^2 + (z_i - 1)^2)$ is the Rosenbrock function in $\mathbb{R}^n$, (Rubinstein and Kroese 2004). Again, the problem is of the form (14), with $p(\mathbf{z}) = 1/4^n$, $\mathbf{z} \in [-2, 2]^n$ and $H(\mathbf{z}) = 4^n h(\mathbf{z})$. Let $\widetilde{p}(\mathbf{z}) = p(\mathbf{z})$, $a = \lambda$ and $b = \ln(4^n)$. Then, (16) can be written as $\ell(\gamma) = \mathbb{P}_f(-\frac{\ln(U)}{\lambda} - R(\mathbf{Z}) \ge \gamma)$, where

$$
f(\mathbf{x}) = \frac{\prod_{i=1}^n I\{-2 \le z_i \le 2\}}{4^n} \times I\{0 < u < 1\}, \quad \mathbf{x} = (\mathbf{z}, u)',
$$

and $\gamma = 0$ is such that (15) is a tight bound. To estimate ℓ we apply the ADAM algorithm using a transition density $\kappa_t(\mathbf{x}^* \,|\, \mathbf{x})$ with stationary pdf

$$f_t(\mathbf{x}) = \frac{f(\mathbf{x}) I\{-\frac{\ln(u)}{\lambda} - R(\mathbf{z}) \geq \gamma_t\}}{\ell(\gamma_t)}.$$

A move from $\mathbf{X} = (\mathbf{Z}, U)'$ to $\mathbf{X}^* = (\mathbf{Z}^*, U^*)'$ uses Gibbs sampling as follows. Given a state $\mathbf{X} = (\mathbf{Z}, U)'$ such that $S(\mathbf{X}) \geq \gamma_t$, we generate $U^* \sim f_t(u \,|\, \mathbf{Z})$. Then, for each $j = 1, \ldots, n-1$ we generate $Z_j^* \sim f_t(z_j \,|\, U^*, Z_1^*, \ldots, Z_{j-1}^*, Z_{j+1}, \ldots, Z_n)$. The distribution of each Z_j^* is uniform on the set $\{[r_1, r_2] \cup [r_3, r_4]\} \cap [-2, 2]$, where $r_1 < r_2$, $r_3 < r_4$ are the real roots of a certain quartic equation $a_1 x^4 + a_2 x^3 + a_3 x^2 + a_4 x + a_5 = 0$. Depending on the coefficients, the quartic equation has either 2 or 4 real roots ($r_3 = r_1$ and $r_4 = r_2$). Finally, we generate $Z_n^* \sim f_t(z_n \,|\, U^*, Z_1^*, \ldots, Z_{n-1}^*)$. The random variable Z_n^* has uniform distribution on the set $[r_1, r_2] \cap [-2, 2]$, where $r_1 < r_2$ are the roots of a certain quadratic equation $a_1 x^2 + a_2 x + a_3 = 0$. For more details see Botev (2009).

As a numerical example, consider the case where $\lambda = 10^4$ and $n = 10$. We run the ADAM algorithm 400 independent times with $\varrho = 0.5$ and $N = 10^3$, and obtained $\widehat{\ell} = 9.7 \times 10^{-36}$ with estimated relative error (using the data from the 400 runs) of 7%. Therefore, $\widehat{\mathcal{Z}} = \widehat{\ell} \times \mathrm{e}^{a\gamma + b} = \widehat{\ell} \times 4^{10} \approx 1.0 \times 10^{-29}$ with relative error of 7%. Each run of the ADAM algorithm took about 117 iterations ($T = 117$), giving a total simulation effort of $N \times 400 \times T = 46.8 \times 10^6$ samples. For the same simulation effort the CMC estimator $\frac{1}{M} \sum_{i=1}^{M} \exp(-\lambda R(\mathbf{Z}))$, $M = 46.8 \times 10^6$, with $\{\mathbf{Z}_i\} \sim_{\text{iid}} \mathsf{U}[-2, 2]^{10}$ gives an estimated relative error of 99.9%. To achieve a relative error of 7% using CMC estimation would require a simulation effort of approximately 2×10^{37} samples.

Remark 6.1 (Optimization of Rosenbrock function) Numerical minimization of the Rosenbrock function $R(\mathbf{z})$ is a challenging minimization problem (Rubinstein and Kroese 2004). It is commonly used as a test case for a wide range of numerical optimization routines. The function $R(\mathbf{z})$ has a global minimum of 0 at $\mathbf{z} = (1, \ldots, 1)'$. One way in which $R(\mathbf{z})$ could be minimized is to sample approximately from the Boltzmann density $\mathrm{e}^{-\lambda R(\mathbf{z})}/\mathcal{Z}$, $\mathbf{z} \in [-2, 2]^n$ for a large value of λ. In Example 6.2, as a consequence of estimating the constant $\mathcal{Z}$ using Algorithm 6.1, we also obtain an estimate for the global minimizer of $R(\mathbf{z})$. In particular, the population $\mathcal{X}_T = \{\mathbf{X}_1, \ldots, \mathbf{X}_{N_T}\}$ at the final iteration of the ADAM algorithm is approximately distributed according to the stationary density $f_T(\mathbf{x})$. Hence, $\mathbf{Z}_i$ in $\mathbf{X}_i = (\mathbf{Z}_i, U_i)'$ is approximately distributed according to the marginal (Boltzmann density) $p(\mathbf{z})H(\mathbf{z})/\mathcal{Z} = \mathrm{e}^{-\lambda R(\mathbf{z})}/\mathcal{Z}$, and we can use $\mathbf{Z}_i : i = \operatorname{argmin}_j R(\mathbf{Z}_j)$ as an estimate for the global minimizer of $R(\mathbf{z})$. For the numerical example considered above we obtain

$$\mathbf{Z}_i = (1.00,\ 0.99,\ 0.99,\ 0.99,\ 1.00,\ 1.00,\ 1.00,\ 1.00,\ 1.00,\ 1.00)'$$

with $R(\mathbf{Z}_i) \approx 5 \times 10^{-5}$. The result is close to the true minimizer $(1, \ldots, 1)'$. We obtain similar results for $n = 100$. This example illustrates how we can use Algorithm 6.1 (with A set to be ADAM) as an optimization algorithm. This is similar to the *simulated annealing* algorithm (Liu 2001; Rubinstein and Kroese 2007), in which the MH sampler is used to approximately sample from the Boltzmann density and minimize the function $R(\mathbf{z})$. For more optimization examples see Botev (2009).

7 MCMC sampling

In this section we consider using the GS Algorithm 3.1 as an alternative to standard Markov chain Monte Carlo sampling from multidimensional pdfs of the form:

$$f_t(\mathbf{x}) = \frac{f(\mathbf{x})\, I_{\{S(\mathbf{x}) \geq \gamma_t\}}}{\ell(\gamma_t)}. \tag{18}$$

Note that since (17) can be viewed as a marginal density of (18) for $t = T$, sampling from (17) can be achieved by sampling from (18). We show that the population $\mathcal{X}_T$ in the final stage of the GS algorithm can be treated as a sample from the multidimensional pdf (18) even in cases where standard Markov chain Monte Carlo algorithms are impractical due to poor mixing. In addition, we also provide a convergence diagnostic which tests the hypothesis that the population $\mathcal{X}_T$ is drawn from the target pdf (18). Deciding when a Markov chain has converged is an important problem in applications of Markov chain Monte Carlo. Many methods for diagnosing convergence have been proposed, ranging from simple graphical methods to computationally intensive hypothesis tests (Brooks et al. 1997; Brooks and Roberts 1998). See also Del Moral et al. (2006) for a similar SMC approach to diagnosing convergence of MCMC algorithms.

For clarity of presentation we explain how to sample from (18) in a separate algorithm, in which the transition density is *reversible*.

Algorithm 7.1 (Splitting Sampler) Given a sequence $\{\gamma_t, \varrho_t\}_{t=1}^T$, set $s_t = \lceil \varrho_{t+1}^{-1} \rceil$ for all $t < T$ and execute the following steps.

1. *Initialize.* Set the counter $t = 1$. Keep generating $\mathbf{X} \sim f(\mathbf{x})$, until $S(\mathbf{X}) \geq \gamma_1$. Let $\mathbf{X}^1 = \mathbf{X}$ be the output. Note that $\mathbf{X}^1$ has density $f_1(\mathbf{x}) = f(\mathbf{x})\, I_{\{S(\mathbf{x}) \geq \gamma_1\}}/c_1$.

2. *Markov chain sampling.* Generate

$$\mathbf{Y}_j \sim_{\text{iid}} \kappa_t(\mathbf{y}\,|\,\mathbf{X}^t), \quad j = 1,\ldots,s_t, \tag{19}$$

where $\kappa_t(\mathbf{y}\,|\,\mathbf{X}^t)$ is a *reversible* Markov transition density with stationary pdf $f_t(\mathbf{y})$. Let $N_{t+1} = \sum_{j=1}^{s_t} I_{\{S(\mathbf{Y}_j)\geq\gamma_{t+1}\}}$. If $N_{t+1} = 0$, repeat from Step 1; otherwise, continue with Step 3.

3. *Updating.* Let $\mathbf{X}^{t+1}$ be a uniformly sampled point from the set of points $\{\mathbf{Y}_1,\ldots,\mathbf{Y}_{s_t}\}$ such that $S(\mathbf{X}^{t+1}) \geq \gamma_{t+1}$. The pdf of $\mathbf{X}^{t+1}$ is thus given by the conditional density

$$\frac{I_{\{S(\mathbf{x}^{t+1})\geq\gamma_{t+1}\}}\,\kappa_t(\mathbf{x}^{t+1}\,|\,\mathbf{X}^t)}{c_{t+1}(\mathbf{X}^t)}, \tag{20}$$

where $c_{t+1}(\mathbf{y}) = \int I_{\{S(\mathbf{x})\geq\gamma_{t+1}\}}\,\kappa_t(\mathbf{x}\,|\,\mathbf{y})\,\mathrm{d}\mathbf{x}$ is the probability that a move of the Markov chain starting in state $\mathbf{y}$ has a performance above γ_{t+1}. Note that an unbiased estimate of $c_{t+1}(\mathbf{X}^t)$ is $\widehat{c}_{t+1}(\mathbf{X}^t) = \frac{N_{t+1}}{s_t}$, so that $\mathbb{E}[\widehat{c}_{t+1}(\mathbf{X}^t)\,|\,\mathbf{X}^t] = c_{t+1}(\mathbf{X}^t)$. Reset the counter $t = t+1$.

4. *Final Output.* If $t = T$, output $\{\widehat{c}_{t+1}(\mathbf{X}^t)\}_{t=1}^{T-1}$ and $(\mathbf{X}^1,\ldots,\mathbf{X}^T)$; otherwise, repeat from Step 2.

A diagnostic test is based on the following result.

Proposition 7.1 (Sufficient Condition for Stationarity) *If* $\sum_{t=1}^{T-1}\ln c_{t+1}(\mathbf{x}^t)$ *is a constant for every* $(\mathbf{x}^1,\ldots,\mathbf{x}^T)$ *such that* $S(\mathbf{x}^t)\geq\gamma_t$, $t = 1\ldots,T-1$, *then the final state* $\mathbf{X}^T$ *of Algorithm 7.1 has the pdf*

$$f_T(\mathbf{x}) = \frac{I_{\{S(\mathbf{x})\geq\gamma\}}\,f(\mathbf{x})}{\ell(\gamma)}.$$

In other words, if $\sum_{t=1}^{T-1}\ln c_{t+1}(\mathbf{x}^t)$ *does not depend on* $(\mathbf{x}^1,\ldots,\mathbf{x}^T)$, *then the Markov chain of Algorithm 7.1 is in stationarity.*

Proof First, the joint pdf of $(\mathbf{X}^1,\ldots,\mathbf{X}^T)$ is:

$$\widehat{f}_T(\mathbf{x}^1,\ldots,\mathbf{x}^T) = \frac{f(\mathbf{x}^1)\,I_{\{S(\mathbf{x}^1)\geq\gamma_1\}}}{c_1}\prod_{t=1}^{T-1}\frac{I_{\{S(\mathbf{x}^{t+1})\geq\gamma_{t+1}\}}\,\kappa_t(\mathbf{x}^{t+1}\,|\,\mathbf{x}^t)}{c_{t+1}(\mathbf{x}^t)}.$$

Using the reversibility of the transition densities $\{\kappa_t\}$, we can write the joint pdf as

$$\widehat{f}_T(\mathbf{x}^1,\ldots,\mathbf{x}^T) = \frac{f(\mathbf{x}^T)\,I_{\{S(\mathbf{x}^T)\geq\gamma_T\}}}{c_1}\prod_{t=1}^{T-1}\frac{\kappa_t(\mathbf{x}^t\,|\,\mathbf{x}^{t+1})}{c_{t+1}(\mathbf{x}^t)}. \tag{21}$$

Ideally, we would like the joint pdf in (21) to be identical to the target:

$$f_T(\mathbf{x}^1,\ldots,\mathbf{x}^T) = \frac{f(\mathbf{x}^T)\,I_{\{S(\mathbf{x}^T)\geq\gamma_T\}}}{\ell}\prod_{t=1}^{T-1}\kappa_t(\mathbf{x}^t\,|\,\mathbf{x}^{t+1}), \tag{22}$$

because then $\mathbf{x}^T$ has the desired marginal density $f_T(\mathbf{x})$. We can measure how close the sampling density $\widehat{f}_T(\mathbf{x}^1,\ldots,\mathbf{x}^T)$ is from the target density $f_T(\mathbf{x}^1,\ldots,\mathbf{x}^T)$ using any distance measure from the Csisár's *ϕ-divergence* family of measures (Botev and Kroese 2009; Rubinstein and Kroese 2007). A convenient member of Csisár's family of measures is the χ^2 *goodness of fit* divergence defined as $\mathcal{D}(p\to q) = \frac{1}{2}\int\frac{(p(\mathbf{x})-q(\mathbf{x}))^2}{p(\mathbf{x})}\,d\mathbf{x} = -\frac{1}{2}+\frac{1}{2}\mathbb{E}\frac{q^2(\mathbf{X})}{p^2(\mathbf{X})}$, for any given pair of pdfs p and q. Thus, we can measure the closeness between the sampling pdf (21) and the target pdf (22) via $\mathcal{D}(\widehat{f}_T\to f_T) = -\frac{1}{2}+\frac{1}{2}\mathbb{E}_{\widehat{f}_T}\prod_{t=1}^{T-1}\frac{c_{t+1}^2(\mathbf{X}^t)}{c_{t+1}^2}$. Hence, after rearranging, we have

$$2\,\frac{\ell^2}{c_1^2}\,\mathcal{D}(\widehat{f}_T\to f_T) = \mathbb{E}_{\widehat{f}_T}\prod_{t=1}^{T-1}c_{t+1}(\mathbf{X}^t) - \frac{\ell^2}{c_1^2} = \mathrm{Var}_{\widehat{f}_T}\left(\prod_{t=1}^{T-1}c_{t+1}(\mathbf{X}^t)\right),$$

where we have used the fact that $\mathbb{E}_{\widehat{f}_T}\prod_{t=1}^{T-1}c_{t+1}(\mathbf{X}^t) = \ell/c_1$. It follows that the distance between (21) and (22) is zero if and only if $\mathrm{Var}_{\widehat{f}_T}\prod_{t=1}^{T-1}c_{t+1}(\mathbf{X}^t) = 0$. In other words, if $\prod_{t=1}^{T-1}c_{t+1}(\mathbf{X}^t)$ (or $\sum_{t=1}^{T-1}\ln c_{t+1}(\mathbf{X}^t)$) is a constant, then the pdfs (21) and (22) are identical and $\mathbf{X}^T$ has the desired marginal pdf. □

Algorithm 7.2 uses an ANOVA to test if $\sum_{t=1}^{T-1}\ln c_{t+1}(\mathbf{X}^t)$ is a constant.

Algorithm 7.2 (χ^2 diagnostic test for stationarity with respect to $f_T(\mathbf{x})$)

1. Let $(\mathbf{X}_1^1,\ldots,\mathbf{X}_1^T),\ldots,(\mathbf{X}_M^1,\ldots,\mathbf{X}_M^T) \sim \widehat{f}_T(\mathbf{x}^1,\ldots,\mathbf{x}^T)$ be a population from the sampling density (21) obtained via Algorithm 7.1, and let C be the $M\times(T-1)$ matrix of estimates:

$$C = \begin{bmatrix} \ln\widehat{c}_2(\mathbf{X}_1^1) & \ln\widehat{c}_3(\mathbf{X}_1^2) & \cdots & \ln\widehat{c}_T(\mathbf{X}_1^{T-1}) \\ \ln\widehat{c}_2(\mathbf{X}_2^1) & \ln\widehat{c}_3(\mathbf{X}_2^2) & \cdots & \ln\widehat{c}_T(\mathbf{X}_2^{T-1}) \\ \vdots & \vdots & \cdots & \vdots \\ \ln\widehat{c}_2(\mathbf{X}_M^1) & \ln\widehat{c}_3(\mathbf{X}_M^2) & \cdots & \ln\widehat{c}_T(\mathbf{X}_M^{T-1}) \end{bmatrix},$$

where the i-th row depends on $(\mathbf{X}_i^1,\mathbf{X}_i^2,\ldots,\mathbf{X}_i^T)$.

2. Compute the following statistics:

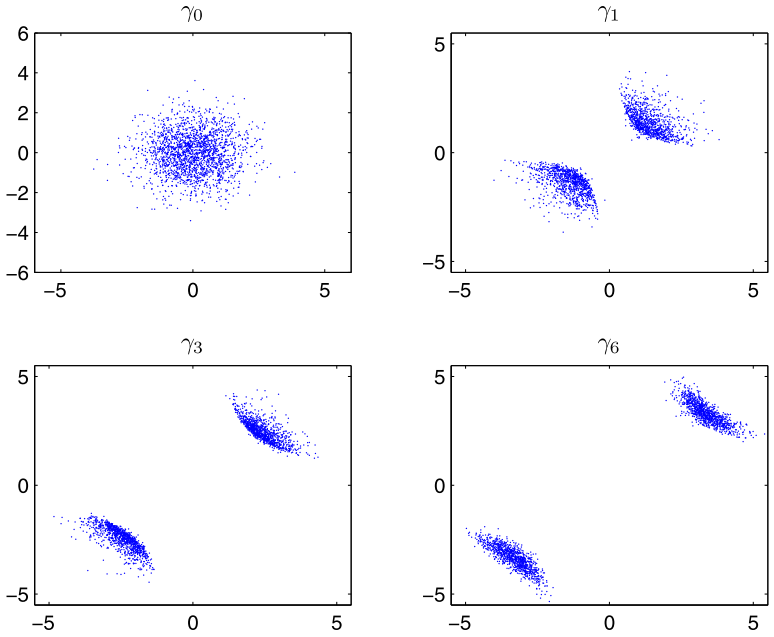

Fig. 5 The empirical distribution of $\mathbf{Z}$ conditional on $S(\mathbf{X}) \geq \gamma_t$ for $t = 0, 1, 3, 6$, respectively

row means: $\bar{C}_{i\bullet} = \dfrac{1}{T-1}\sum_{j=1}^{T-1} C_{ij}$,

column means: $\bar{C}_{\bullet j} = \dfrac{1}{M}\sum_{i=1}^{M} C_{ij}$

overall mean: $\bar{C} = \dfrac{1}{T-1}\sum_{j=1}^{T-1} \bar{C}_{\bullet j}$,

"row effect" sum of squares: $\mathrm{SSTR} = \sum_{i=1}^{M}(\bar{C}_{i\bullet} - \bar{C})^2$

"within row" variance:

$$\mathrm{SSE} = \frac{1}{(M-1)(T-1)^2}\sum_{j=1}^{T-1}\sum_{i=1}^{M}(C_{ij} - \bar{C}_{\bullet j})^2.$$

3. Under the hypothesis that $\sum_{t=1}^{T-1} \ln(c_{t+1}(\mathbf{X}^t))$ is a constant, and assuming an approximately normal distribution for $\{\bar{C}_{i\bullet}, \bar{C}_{\bullet j}\}$, the test statistic $\mathcal{T} = \frac{\mathrm{SSTR}}{\mathrm{SSE}}$ has the χ^2 distribution with $M-1$ degrees of freedom. The p-value (of a χ^2 test to diagnose convergence of the Markov chains, see Gelman and Rubin 1992) is $1 - \Phi(\mathcal{T})$, where Φ is the cdf of the χ^2 distribution with $M-1$ degrees of freedom.

If the chain in Algorithm 7.1 samples according to the target, then the sums across each row of matrix C should be roughly the same. The two-way *analysis-of-variance* test in Algorithm 7.2 simply tests for row effects in matrix C. Each column of C represents a different level, while each row of C represents a given factor. We caution that most diagnostics frequently successfully detect undesirable Markov chain behavior (slow mixing or lack of stationarity), but they can never be used to demonstrate in any meaningful way that the Markov chain accurately samples from the target pdf.

Example 7.1 (Comparison with Gibbs Sampler) To illustrate the performance of the splitting sampler, we consider the problem of sampling from the pdf in Example 6.1. We run Algorithm 6.1 with exactly the same setup as in Example 6.1, except that the three steps in Algorithm 6.2 are executed in a random order, resulting in random Gibbs sampling, as opposed to systematic Gibbs sampling. The random Gibbs sampling ensures that the transition density $\kappa_t(\mathbf{X}^* \,|\, \mathbf{X})$ is reversible. Figure 5 shows the empirical distribution of $\mathbf{Z}$ at levels $(\gamma_0, \gamma_1, \gamma_3, \gamma_6) = (-\infty, -117.91, -44.78, 0)$. The $\gamma_0 = -\infty$ case shows the sample from the proposal $p(\mathbf{z})$, and the γ_6 case shows 2030 points approximately distributed from the target density given in the left panel of Fig. 6. Notice how the two distinct modes emerge gradually. The p-value in Step 3 of Algorithm 7.2 is 0.1, thus failing to detect transient behavior and supporting the hypothesis that the chain samples according to the target. In addition, the proportion of points in each mode at the final stage is roughly equal to a half, namely, 1009 points belong to the upper right mode and 1021 points belong to the lower left mode. Note that the standard Gibbs sampler applied to $h(\mathbf{z}) = \exp(-(z_1^2 + z_2^2 + (z_1 z_2)^2 - 2\lambda z_1 z_2)/2)$, $\lambda = 12$ fails. In particular, starting from $(0, 0)$ we iterate the following steps 10^9 times.

- Given (Z_1, Z_2), generate $Z_1^* \sim \mathsf{N}(\frac{\lambda Z_2}{1+Z_2^2}, \frac{1}{Z_2^2+1})$.
- Given Z_1^*, generate $Z_2^* \sim \mathsf{N}(\frac{\lambda Z_1^*}{1+(Z_1^*)^2}, \frac{1}{(Z_1^*)^2+1})$. Update $(Z_1, Z_2) = (Z_1^*, Z_2^*)$.

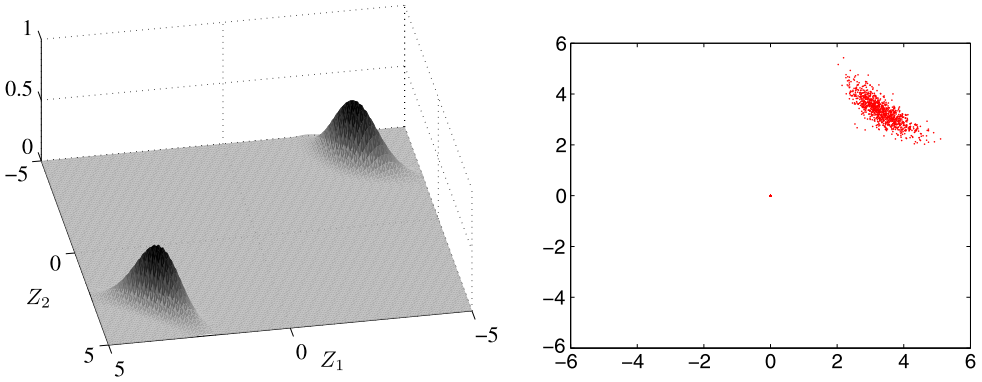

Fig. 6 *Left panel*: Plot of the two-humps density. *Right panel*: Empirical distribution of the output of the standard Gibbs sampler

The right panel of Fig. 6 shows that the standard Gibbs sampler results in a chain which is trapped in one of the two modes and fails to mix satisfactorily in 10^9 steps. Here the chain of length 10^9 is thinned to have 10^3 points, that is, we keep the 10^6-th, 2×10^6-th, 3×10^6-th, etc. points from the original Markov chain sequence. Our numerical experience suggests that the performance of the Gibbs sampler is affected by the starting value of the chain. In contrast, the problem of selecting starting values for the chains within the splitting sampler is strongly mitigated. Overall, the splitting sampler explores the support of the target density better than the standard Gibbs sampler. For a comparison with other samplers such as the *equi-energy* sampler (Kou et al. 2006), we refer to Botev (2009).

8 Conclusions and suggestions for future work

This paper presents a Generalized Splitting algorithm, that extends the original splitting idea of Kahn and Harris to static and non-Markovian simulation problems. Similar to the original splitting method, the GS method induces a branching process by constructing an artificial Markov chain via the Gibbs or Hit-and-Run samplers. Notable features of the proposed approach are as follows.

First, the GS algorithm provides an unbiased and consistent estimator of ℓ in (1) without requiring that the Markov chain constructed by the GS algorithm reaches stationarity. It is not necessary that the chain is irreducible. In contrast, standard MCMC algorithms provide a biased estimate of ℓ for any finite run time. In general, this bias can only be reduced by discarding observations during the initial transient or burn-in phase of the MCMC algorithms. Thus, any inference is always based upon a portion of the sampler output. In addition, the GS algorithm provides an unbiased estimate of the mean square error of $\widehat{\ell}$. In contrast, standard MCMC algorithms provide biased error estimates.

Second, the GS algorithm can be used to generate samples (approximately) according to a given multidimensional pdf, for which standard MCMC methods fail, by significantly improving the exploration of the multidimensional pdf. In addition, the GS-sampler provides a computationally inexpensive convergence diagnostic based on a χ^2 test statistic. In contrast, most of the existing convergence diagnostics (Brooks et al. 1997) are computationally intensive and graphical in nature.

A possible direction for future research includes the classical problem in splitting: finding an optimal importance function for a given rare-event estimation problem. That is, we aim to estimate $\mathbb{P}(h_{t+1}(\mathbf{X}) \geq \gamma \,|\, h_t(\mathbf{X}) \geq \gamma)$ for an optimal (in minimal variance sense) sequence of importance functions $\{h_t\}_{t=0}^T$, where $h_T(\mathbf{X}) = S(\mathbf{X})$ and the events $\{h_t(\mathbf{X}) \geq \gamma\} \subseteq \{h_{t+1}(\mathbf{X}) \geq \gamma\}$ for all t. In this paper we have only considered the quite arbitrary special case where $h_t(\mathbf{X}) = S(\mathbf{X})/a_t$ (that is, $\mathbb{P}(S(\mathbf{X}) \geq \gamma_{t+1} \,|\, S(\mathbf{X}) \geq \gamma_t)$ with $\gamma_t = a_t \gamma$) for some suitably chosen sequence $\{a_t\}$.

References

Botev, Z.I.: Three examples of a practical exact Markov chain sampling. Postgraduate Seminar Paper, School of Mathematics and Physics, The University of Queensland. http://espace.library.uq.edu.au/view/UQ:130865 (2007)

Botev, Z.I.: An algorithm for rare-event probability estimation using the product rule of probability theory. Technical report, School of Mathematics and Physics, The University of Queensland. http://espace.library.uq.edu.au/view/UQ:151299 (2008)

Botev, Z.I.: Splitting methods for efficient combinatorial counting and rare-event probability estimation. Technical report, School of Mathematics and Physics, The University of Queensland. http://espace.library.uq.edu.au/view/UQ:178513 (2009)

Botev, Z.I., Kroese, D.P.: The generalized cross entropy method, with applications to probability density estimation. Methodol. Comput. Appl. Probab. (2009). doi:10.1007/s11009-009-9133-7

Brooks, S.P., Roberts, G.O.: Convergence assessment techniques for Markov Chain Monte Carlo. Stat. Comput. **8**, 319–335 (1998)

Brooks, S.P., Dellaportas, P., Roberts, G.O.: An approach to diagnosing total variation convergence of MCMC algorithms. J. Comput. Graph. Stat. **1**, 251–265 (1997)

Cérou, F., Guyader, A.: Adaptive multilevel splitting for rare event analysis. Stoch. Anal. Appl. **25**, 417–443 (2007)

Cérou, F., Del Moral, P., Furon, T., Guyader, A.: Rare-event simulation for a static distribution. INRIA-00350762 (2009)

Chen, M.H., Shao, Q.M., Ibrahim, J.G.: Monte Carlo Methods in Bayesian Computation. Springer, New York (2000)
Gander, W., Gautschi, W.: Adaptive quadrature—revisited. BIT Numer. Math. **40**, 84–101 (2000)
Garvels, M.J.J.: The splitting method in rare event simulation. PhD thesis, University of Twente, The Netherlands, October 2000
Garvels, M.J.J., Kroese, D.P.: A comparison of RESTART implementations. In: Proceedings of the 1998 Winter Simulation Conference, pp. 601–609. Washington, DC (1998)
Garvels, M.J.J., Kroese, D.P., van Ommeren, J.C.W.: On the importance function in splitting simulation. Eur. Trans. Telecommun. **13**(4), 363–371 (2002)
Gelman, A., Rubin, D.: Inference from iterative simulation using multiple sequences (with discussion). Stat. Sci. **7**, 457–511 (1992)
Glasserman, P., Heidelberger, P., Shahabuddin, P., Zajic, T.: A look at multilevel splitting. In: Niederreiter, H. (ed.) Monte Carlo and Quasi Monte Carlo Methods 1996. Lecture Notes in Statistics, vol. 127, pp. 99–108. Springer, New York (1996)
Glasserman, P., Heidelberger, P., Shahabuddin, P., Zajic, T.: A large deviations perspective on the efficiency of multilevel splitting. IEEE Trans. Autom. Control **43**(12), 1666–1679 (1998)
Gu, J., Purdom, P.W., Franco, J., Wah, B.W.: Algorithms for the satisfiability (SAT) problem: a survey. In: Satisfiability Problem: Theory and Applications. DIMACS Series in Discrete Mathematics, vol. 35. American Mathematical Society, Providence (1996)
Hoos, H.H., Stützle, T.: SATLIB: an online resource for research on SAT. In: Gent, I.P., Maaren, H.V., Walsh, T. (eds.) SAT 2000, pp. 283–292. IOS Press, The Netherlands (2000). www.satlib.org
Johansen, A.M., Del Moral, P., Doucet, A.: Sequential Monte Carlo samplers for rare events. In: Proc. 6th International Workshop on Rare Event Simulation (2006)
Kahn, H., Harris, T.E.: Estimation of Particle Transmission by Random Sampling. National Bureau of Standards Applied Mathematics Series (1951)
Kou, S.C., Zhou, Q., Wong, W.H.: Equi-energy sampler with applications in statistical inference and statistical mechanics. Ann. Stat. **34**, 1581–1619 (2006)
Lagnoux-Renaudie, A.: Rare event simulation. Probab. Eng. Inf. Sci. **20**(1), 45–66 (2006)
Lagnoux-Renaudie, A.: Rare event simulation: effective splitting model under cost constraint. In: Stochastic Processes and Their Applications, pp. 1820–1851 (2008)
L'Ecuyer, P., Demers, V., Tuffin, B.: Splitting for rare-event simulation. In: Proceedings of the 2006 Winter Simulation Conference, pp. 137–148 (2006)
L'Ecuyer, P., Demers, V., Tuffin, B.: Rare events, splitting, and quasi-Monte Carlo. ACM Trans. Model. Comput. Simul. **17**(2), 1–44 (2007)
Liu, J.S.: Monte Carlo Strategies in Scientific Computing. Springer, New York (2001)
Del Moral, P.: Feynman-Kac Formulae: Genealogical and Interacting Particle Systems with Applications. Springer, New York (2004)
Del Moral, P., Doucet, A., Jasra, A.: Sequential Monte Carlo samplers. J. R. Stat. Soc. B **68**(3), 411–436 (2006)
Robert, C.P., Casella, G.: Monte Carlo Statistical Methods, 2nd edn. Springer, New York (2004)
Rubinstein, R.Y., Kroese, D.P.: The Cross-Entropy Method. Springer, New York (2004)
Rubinstein, R.Y., Kroese, D.P.: Simulation and the Monte Carlo Method, 2nd edn. Wiley, New York (2007)
Villén-Altamirano, M., Villén-Altamirano, J.: RESTART: a method for accelerating rare event simulations. In: Cohen, J.W., Pack, C.D. (eds.) Proceedings of the 13th International Teletraffic Congress, Queueing, Performance and Control in ATM, pp. 71–76 (1991)
Villén-Altamirano, M., Villén-Altamirano, J.: RESTART: a straightforward method for fast simulation of rare events. In: Tew, J.D., Manivannan, S., Sadowski, D.A., Seila, A.F. (eds.) Proceedings of the 1994 Winter Simulation Conference, pp. 282–289 (1994)
Villén-Altamirano, M., Villén-Altamirano, J.: About the efficiency of RESTART. In: Proceedings of the RESIM'99 Workshop, pp. 99–128. University of Twente, The Netherlands (1999)
Welsh, D.J.A.: Complexity: Knots, Coloring and Counting. Cambridge University Press, Cambridge (1993)